# Package: mds (via r-universe)

September 7, 2024

**Type** Package

**Title** Medical Devices Surveillance

**Version** 0.3.2

**Maintainer** Gary Chung <gchung05@gmail.com>

**Description** A set of core functions for handling medical device event data in the context of post-market surveillance, pharmacovigilance, signal detection and trending, and regulatory reporting. Primary inputs are data on events by device and data on exposures by device. Outputs include: standardized device-event and exposure datasets, defined analyses, and time series.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.0

**Depends** R (>= 2.10)

**Imports** stats, parsedate, lubridate

**Suggests** testthat, knitr, rmarkdown

**VignetteBuilder** knitr

**Repository** https://gchung05.r-universe.dev

**RemoteUrl** https://github.com/gchung05/mds

**RemoteRef** HEAD

**RemoteSha** 51e88beae68b16ceebad79d2364199396a5d4403

## Contents

---

`define_analyses` *Assess Analyses Definitions*

---

## Description

Define analyses based on an MD-PMS device-event data frame and, optionally, an MD-PMS exposure data frame. See Details for how to use.

## Usage

```
define_analyses(
  deviceevents,
  device_level,
  event_level = NULL,
  exposure = NULL,
  date_level = "months",
  date_level_n = 1,
  covariates = "_none_",
  times_to_calc = NULL,
  invivo = FALSE,
  prior = NULL
)
```

## Arguments

| | |
|---|---|
| `deviceevents` | A device-events object of class `mds_de`, created by a call to `deviceevent()`. |
| `device_level` | String value indicating the source device variable name to analyze by. If `exposure` is specified, `exposure` data will be matched by `device_level`. If a hierarchy of 2 or more are present, see Details for important information.<br>Example: If the `deviceevents` variable column is `device_1` where the source variable name for `device_1` is `'Device Code'`, specify `device_level='Device Code'`. |
| `event_level` | String value indicating the source event variable name to analyze by. Note that `event_level` is not matched to `exposure`. If a hierarchy of 2 or more are present, see Details for important information.<br>Example: If the `deviceevents` variable column is `event_1` where the source variable name for `event_1` is `'Event Code'`, specify `event_level='Event Code'`.<br>Default: `NULL` will not analyze by event. |

| | |
|---|---|
| exposure | Optional exposure object of class mds_e. See details for how exposure analyses definitions are handled.<br>Default: NULL will not consider inclusion of exposure. |
| date_level | String value for the primary date unit to analyze by. Can be either 'months' or 'days'.<br>Default: 'months' |
| date_level_n | Numeric value indicating the number of date_levels to analyze by.<br>Example: date_level='months' and date_level_n=3 indicates analysis on a quarterly level.<br>Default: 1 |
| covariates | Character vector specifying names of covariates to also define analyses for. Acceptable names are covariate variables specified in deviceevents. If the covariate is a factor, additional subgroup analyses will be defined at each level of the factor. "_none_" specifies no covariates, while "_all_" are all covariates specified in deviceevents. See details for more.<br>Example: c("Country", "Region")<br>Default: "_none_" specifies no covariates. |
| times_to_calc | Integer value indicating the number of date units counting backwards from the latest date to define analyses for. If prior is specified, times_to_calc will be ignored.<br>Example 1: times_to_calc=12 with date_level="months" and date_level_n=1 defines analyses for the last year by month.<br>Example 2: times_to_calc=8 with date_level="months" and date_level_n=3 defines analyses for the 2 years by quarter.<br>Default: NULL will define analyses across all available time. |
| invivo | Logical value indicating whether to include time_invivo from deviceevents in the analysis definition. See details for more.<br>Default: FALSE will not include time_invivo in the analysis definition. |
| prior | Future placeholder, currently not used. |

## Details

define_analyses() is a prerequisite to calling time_series(). This function enumerates all possible analyses based on input device-event (deviceevent()) and, optionally, exposure (exposure()) data frames. An analysis is defined as a set of instructions specifying at minimum the device level, event level, the date range of analysis, and the date unit. Additional instructions include the covariate level, time in-vivo status, and exposure levels.

By separating the analysis enumeration (define_analyses()) from the generation of the time series (time_series()), the user may rerun the analyses on different datasets and/or filter the analyses to only those of interest.

The analyses definitions will always include rollup levels for each of device_level, event_level (if specified), and covariates. Rollups are analyses at all device, event, and/or covariate levels. These rollup analyses will be indicated by the keyword 'All' in the analysis definition.

When a hierarchy of 2 or more variables for either device_level or event_level are present in deviceevents, define_analyses() will enforce the 1-level-up parent level ONLY. Additional

higher parent levels are not currently enforced, thus the user is advised to uniquely name the 1-level-up parent level. The parent level DOES NOT ROLLUP currently because the parent level is intended to separate disparate data and devices. This may change in the future.

If `exposure` is specified, any available `match_levels` will be used to calculate the appropriate timeframe for analyses. The exception are the special rollup analyses (see prior paragraph).

When `covariates` are specified, a special rollup analysis definition will always be defined that does not consider the covariates at all. This analysis can be identified by `covariate='Data'` and `covariate_level='All'` in the output `mds_da` object.

When `covariates` are specified and there is no variation in the distribution of covariate values (e.g. all males, all 10, all missing) in the device- and event-specific dataset, these specific analyses will be dropped.

When factor `covariates` are specified, covariate-level analyses may be defined two ways: 1) detect an overall covariate level effect, also known as a 3-dimensional analysis, and 2) subset the data by each level of the covariate, also known as a subgroup analysis. 1) will be denoted as `covariate_level='All'` in the output `mds_da` object, while 2) will specify the factor level in `covariate_level`.

If `invivo=TRUE`, `define_analyses()` will first verify if data exists in the `time_invivo` variable for the given `device_level`, `event_level`, and, if applicable, `covariates` level. If no data exists, `invivo` will be implicitly assigned to `FALSE`.

## Value

A list of defined analyses of class `mds_das`. Each list item, indexed by a numeric key, defines a set of analyses for a unique combination of device, event, and covariate level. Each list item is of the class `mds_da`. Attributes of class `mds_das` are as follows:

**date_level** Defined value for `date_level`

**date_level_n** Defined value for `date_level_n`

**device_level** Defined value for `device_level`

**event_level** Defined value for `event_level`

**times_to_calc** Defined value for `times_to_calc`

**prior_used** Boolean for whether `prior` was specified.

**timestamp** System time when the analyses were defined.

## Examples

```
# Device-Events
de <- deviceevent(
  data_frame=maude,
  time="date_received",
  device_hierarchy=c("device_name", "device_class"),
  event_hierarchy=c("event_type", "medical_specialty_description"),
  key="report_number",
  covariates=c("region"),
  descriptors="_all_")
# Exposures
ex <- exposure(
```

```
  data_frame=sales,
  time="sales_month",
  device_hierarchy="device_name",
  match_levels="region",
  count="sales_volume")
# Defined Analyses - Simple example
da <- define_analyses(de, "device_name")
# Defined Analyses - Simple example with a quarterly analysis
da <- define_analyses(de, "device_name", date_level_n=3)
# Defined Analyses - Example with event type, exposures, and covariates
da <- define_analyses(de, "device_name", "event_type", ex, covariates="region")
```

---

`define_analyses_dataframe` *Create Data Frame from Analyses Definitions*

---

## Description

Returns a data frame summarizing all defined analyses from the `mds_das` object.

## Usage

```
define_analyses_dataframe(inlist)
```

## Arguments

| | |
|---|---|
| `inlist` | Object of class `mds_das` |

## Value

A data frame with each row representing an analysis.

---

`deviceevent` *MD-PMS Device Event Data Frame*

---

## Description

Converts a data frame into a MD-PMS Device Event data frame.

## Usage

```
deviceevent(
  data_frame,
  time,
  device_hierarchy,
  event_hierarchy,
  key = NULL,
  covariates = NULL,
  descriptors = NULL,
  time_invivo = NULL
)
```

## Arguments

| | |
|---|---|
| data_frame | The input data frame requiring components specified in the remaining arguments. |
| time | Character name of date variable in data_frame corresponding to the event. Class must be Date, POSIXt, or character.<br>Example: "event_date" |
| device_hierarchy | Vector of character variable names representing the device hierarchy in data_frame. Vector ordering is lowest level first, most general level last. If more than 2 variables, see important note in Details.<br>Example: c("Version", "Device", "ProductLine") |
| event_hierarchy | Vector of character variable names representing the event hierarchy in data_frame. Vector ordering is most specific event category first, most broad event category last. If more than 2 variables, see important note in Details.<br>Example: c("Event Code", "Event Group") |
| key | Character name of (uniquely identifying) primary key variable in data_frame. Class must be character or numeric.<br>Example: "key_ID"<br>Default: NULL will create a key variable. |
| covariates | Vector of character variable names representing the desired covariates to retain, all of which must be of class numeric or factor. "_all_" includes all covariates, assumed to be remaining variables in data_frame not already specified in key, time, device_hierarchy, or event_hierarchy. Covariates must be numeric, categorical, or binary in nature.<br>Example: c("Reporter", "Operation Time", "Country")<br>Default: NULL includes no covariates. |
| descriptors | Vector of character variable names representing additional descriptive variables that will not be used in any analyses but may be recalled or displayed later during individual device-event review. "_all_" includes all remaining variables in data_frame not already specified in key, time, device_hierarchy, event_hierarchy, or covariates. Typical descriptors are free text or high-dimensional categoricals. |

Example: `c("Description", "Unique Device Identifier")`

Default: `NULL` includes no descriptors.

`time_invivo` Character name of numeric variable in `data_frame` representing the time in vivo of the device at the time of the event `time`. See details for more.

IMPORTANT: If a call to `define_analyses()` is planned, `time_invivo` must be in the time units specified collectively by its parameters `date_level` and `date_level_n`.

Example: `"Implanted Months"`. A value of 45 in the variable `data_frame$'Implanted Months'` would indicate 45 units of time elapsed since the device was first in vivo. If `date_level="months"` and `date_level_n=1`, this will be interpreted by `define_analyses()` as 45 months.

Default: `NULL` indicates this variable will not be used.

## Details

When more than 2 variables are specified in either `device_hierarchy` or `event_hierarchy`, it is important to note that a subsequent call to `define_analyses()` currently only utilizes a maximum of 2 variables: the lowest level and the 1-level-up parent. The user may enforce full hierarchy in >2 variable cases by ensuring that the parent values are uniquely named.

`time_invivo` can be thought of more generally as the time of exposure of the device to the subject at the time of the event. The common usage is duration of the implant in the patient at time of event, for an implantable medical device.

## Value

A standardized MD-PMS data frame of class `mds_de`. Rows are deduplicated. Attributes are as follows:

**key** Original variable name for `key`

**time** Original variable name for `time`

**device_hierarchy** Vector of original variable names for `device_hierarchy` with converted variable names correspondingly named.

**event_hierarchy** Vector of original variable names for `event_hierarchy` with converted variable names correspondingly named.

**covariates** Vector of original variable names for `covariates` with converted variable names correspondingly named.

**descriptors** Vector of original variable names for `descriptors` with converted variable names correspondingly named.

## Examples

```
# A barebones dataset
de <- deviceevent(maude, "date_received", "device_name", "event_type")
# With more variables and variable types
de <- deviceevent(
  data_frame=maude,
  time="date_received",
```

```
  device_hierarchy=c("device_name", "device_class"),
  event_hierarchy=c("event_type", "medical_specialty_description"),
  key="report_number",
  covariates=c("region"),
  descriptors="_all_")
```

---

`exposure` *MD-PMS Exposure Data Frame*

---

## Description

Converts a data frame into a MD-PMS Exposure data frame.

## Usage

```
exposure(
  data_frame,
  time,
  device_hierarchy,
  event_hierarchy = NULL,
  key = NULL,
  match_levels = NULL,
  count = NULL
)
```

## Arguments

| | |
|---|---|
| `data_frame` | The input data frame requiring components specified in the remaining arguments. |
| `time` | Character name of date variable in `data_frame`. Class must be Date, POSIXt, or character.<br>Example: `"event_date"` |
| `device_hierarchy` | Vector of character variable names representing the device hierarchy in `data_frame`. Vector ordering is lowest level first, most general level last.<br>Example: `c("Version", "Device", "ProductLine")` |
| `event_hierarchy` | Vector of character variable names representing the event hierarchy in `data_frame`. Vector ordering is most specific event category first, most broad event category last.<br>Example: `c("Family", "Device", "ProductCode")`<br>Default: `NULL` will not include any event hierarchy. |
| `key` | Character name of (uniquely identifying) primary key variable in `data_frame`. Class must be character or numeric.<br>Example: `"key_ID"`<br>Default: `NULL` will create a key variable. |

`match_levels` Vector of character variable names in `data_frame` representing additional grouping factors for exposure. Specified variables will be implicitly matched to equivalently named variables contained in the `mds_de` object class.

Example: `c("Country", "Region")`

Default: `NULL` will not include any additional grouping factors.

`count` Character name of exposure count variable in `data_frame`. Class must be numeric.

Example: `"Units Sold"`

Default: `NULL` will assume each row represents one exposure.

## Value

A standardized MD-PMS data frame of class `mds_e`. Rows are deduplicated. Attributes are as follows:

**key** Original variable name for `key`

**time** Original variable name for `time`

**device_hierarchy** Vector of original variable names for `device_hierarchy` with converted variable names correspondingly named.

**event_hierarchy** Vector of original variable names for `event_hierarchy` with converted variable names correspondingly named.

**match_levels** Vector of variable names for grouping factors

**count** Original variable name for `count`

## Examples

```
# A barebones dataset
ex <- exposure(sales, "sales_month", "device_name")
# With more variables and variable types
ex <- exposure(
  data_frame=sales,
  time="sales_month",
  device_hierarchy="device_name",
  match_levels="region",
  count="sales_volume")
```

---

# maude *Bone Cement MAUDE Events in 2017*

---

## Description

A dataset containing 535 events reported into the FDA MAUDE database on bone cement in 2017. Data were obtained via the openFDA API (https://open.fda.gov).

## Usage

```
maude
```

## Format

A data frame with 535 rows and 15 variables. Full variable descriptions may be found on the FDA Device Reference Guide (https://open.fda.gov). Note that region is a simulated variable not present in MAUDE. Descriptions as follows:

**report_number** Identifying number for the adverse event report.

**event_type** Outcomes associated with the adverse event.

**date_received** Date the report was received by the FDA.

**product_problem_flag** Indicates whether or not a report was about the quality, performance or safety of a device.

**adverse_event_flag** Whether the report is about an incident where the use of the device is suspected to have resulted in an adverse outcome in a patient.

**report_source_code** Source of the adverse event report.

**lot_number** The lot number found on the label or packaging material.

**model_number** The exact model number found on the device label or accompanying packaging.

**manufacturer_d_name** Device manufacturer name.

**manufacturer_d_country** Device manufacturer country.

**brand_name** The trade or proprietary name of the suspect medical device as used in product labeling or in the catalog.

**device_name** This is the proprietary name, or trade name, of the cleared device.

**medical_specialty_description** Regulation Medical Specialty is assigned based on the regulation (e.g. 21 CFR Part 888 is Orthopedic Devices).

**device_class** A risk based classification system for all medical devices ((Federal Food, Drug, and Cosmetic Act, section 513)

**region** A simulated, randomly assigned geographical region for package example purposes.

## Source

https://open.fda.gov/data/maude/

---

# mds_ts *Example List of mds_ts Time Series Objects*

---

## Description

An example list of time series objects (class mds_ts) generated using the mds package.

## Usage

```
mds_ts
```

### Format

A list of 3 elements each of class `mds_ts`

### Source

See `?maude` and `?sales` for source device-event and exposure data. See `?mds::time_series` for how to generate `mds_ts` time series.

---

`plot.mds_ts` *Plot MD-PMS Time Series*

---

### Description

Quickly visualizes an MD-PMS times series of class `mds_ts`.

### Usage

```
## S3 method for class 'mds_ts'
plot(x, mode = "nA", xlab = "Time", ylab = "Count", main = NULL, ...)
```

### Arguments

| | |
|---|---|
| `x` | An object of class `mds_ts`. |
| `mode` | Series to plot. Valid values are: `'nA'`, `'nB'`, `'nC'`, `'nD'`, `'exposure'`, `'rate'`. `'rate'` is simply `'nA' / 'exposure'`. See details for more.<br>Default: `'nA'` |
| `xlab` | x-axis label<br>#' Default: `'Time'` |
| `ylab` | y-axis label<br>Default: `'Count'` |
| `main` | Plot title<br>Default: `NULL` infers the title from `x` and `mode`. |
| `...` | Further arguments to pass onto `plot()` generic. |

### Details

`mode` values defined as follows. Note: The following definitions use a device-event pair as a working example, however it may also be a covariate-device pair.

**'nA'** Counts of the device-event pair.

**'nB'** Counts of the device for all other events.

**'nC'** Counts of all other devices for the event.

**'nD'** Counts of all other devices for all other events.

**'exposure'** Counts of exposure for the device-event pair.

**'rate'** A crude rate, calculated as the device-event counts pair divided by the exposure counts.

## sales *Simulated Device Sales Data*

### Description

A dataset containing simulated monthly sales by device and country for devices reported in the maude dataset. For package usage examples, this data serves as a proxy for exposures. Data were generated using a random normal distribution weighted by the number of reported events by device and country.

### Usage

```
sales
```

### Format

A data frame with 360 rows and 4 variables:

**device_name** Name of the device mapped from the maude dataset.

**region** Geographical region mapped from the maude dataset.

**sales_month** The month of sales.

**sales_volume** The volume of sales.

### Source

Random normal distribution using rnorm().

## summary.mds_das *Summarize a Collection of MD-PMS Defined Analyses Prints basic counts and date ranges by various analysis factors as defined in the original* define_analyses() *call.*

### Description

Summarize a Collection of MD-PMS Defined Analyses Prints basic counts and date ranges by various analysis factors as defined in the original define_analyses() call.

### Usage

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

### Arguments

| | |
|---|---|
| object | A MD-PMS Defined Analyses object of class mds_das |
| ... | Additional arguments affecting the summary produced |

### Value

List of analyses counts and date ranges.

---

`time_series` *Generate Time Series from Defined Analysis or Analyses*

---

### Description

Creates time series data frame(s) from defined analysis/analyses (`define_analyses()`), device-event data frame (`deviceevent()`), and optionally, exposure data frame (`exposure()`). If analysis includes covariates or time in-vivo, creates the relevant supporting data frame.

### Usage

```
time_series(analysis, ...)

## S3 method for class 'list'
time_series(analysis, ...)

## S3 method for class 'mds_das'
time_series(analysis, ...)

## S3 method for class 'mds_da'
time_series(analysis, deviceevents, exposure = NULL, use_hierarchy = T, ...)
```

### Arguments

| | |
|---|---|
| `analysis` | A defined analysis object of class `mds_da`, list of class `mds_das`, or a list of objects each of class `mds_da`, usually created by `define_analyses()`. |
| `...` | Further arguments for future work. |
| `deviceevents` | A device-event data frame of class `mds_de`, usually created by `deviceevent()`. This should be the same data frame used to generate `analysis`. |
| `exposure` | Optional exposure data frame of class `mds_e`, usually created by `exposure()`. This should be the same data frame used to generate `analysis`, if exposure data was used.<br>Default: `NULL` will not consider exposure data. |
| `use_hierarchy` | Deprecated - do not use. Logical value indicating whether device and event hierarchies should be used in counting contingency tables for disproportionality analysis. |

### Value

A standardized MD-PMS time series data frame of class `mds_ts`.

The data frame contains, by defined date levels, the following columns:

**nA** Count of the device & event level of interest. If covariate analysis is indicated, this will be at the covariate & device level of interest.

**nB** Optional. Count of the device & non-event, or if covariate analysis, covariate & non-device. `nB` will be missing if this is an `'All'` level analysis.

**nC** Optional. Count of the non-device & event, or if covariate analysis, non-covariate & device. `nC` will be missing if this is an `'All'` level analysis.

**nD** Optional. Count of the non-device & non-event, or if covariate analysis, non-covariate & non-device. `nD` will be missing if this is an `'All'` level analysis.

**ids** List of all keys from `deviceevents` constituting `nA`.

**exposure** Optional. Count of exposures applicable to `nA`. This counts at the device and covariate levels but not at the event level. If a matching device and/or covariate level is not found, then `exposure` will be `NA`. The exception is an `'All'` level analysis, which counts exposures across all levels.

**ids_exposure** Optional. List of all exposure keys from `exposure` applicable to `nA`.

The `mds_ts` class attributes are as follows:

**title** Short description of the analysis.

**analysis** The analysis definition of class `mds_da`.

**exposure** Boolean of whether exposure counts are present.

**dpa** Boolean of whether 2x2 contingency table counts are present (presumably for disproportionality analysis or 'DPA').

**dpa_detail** Optional. If `dpa` is `TRUE`, `list` object containing labels for the DPA contingency table.

**covar_data** Optional. If analysis definition includes covariate level or time in-vivo, `data.frame` object containing the relevant data.

## Methods (by class)

- `list`: Generate time series from a list
- `mds_das`: Generate time series from a list of defined analyses
- `mds_da`: Generate time series using defined analysis

## Examples

```
de <- deviceevent(maude, "date_received", "device_name", "event_type")
ex <- exposure(sales, "sales_month", "device_name", count="sales_volume")
da <- define_analyses(de, "device_name", exposure=ex)
# Time series on one analysis
time_series(da, de, ex)
# Time series on multiple analyses
time_series(da[1:3], de, ex)
```

# Index