

# Package ‘mudata2’

August 20, 2018

**Title** Interchange Tools for Multi-Parameter Spatiotemporal Data

**Version** 1.0.3

**Description** Formatting and structuring multi-parameter spatiotemporal data is often a time-consuming task. This package offers functions and data structures designed to easily organize and visualize these data for applications in geology, paleolimnology, dendrochronology, and paleoclimate. See Dunnington and Spooner (2018) <doi:10.1139/facets-2017-0026>.

**Imports** ggplot2, dplyr (>= 0.7), jsonlite (>= 1.2), tibble, magrittr, stringr, readr, tidyr, lubridate, rlang, tidyselect

**Depends** R (>= 3.2.2)

**Suggests** testthat, RSQLite, dbplyr, sf (>= 0.5.5), covr, knitr, rmarkdown, hms

**License** GPL-2

**URL** <https://github.com/paleolimbot/mudata>

**BugReports** <https://github.com/paleolimbot/mudata/issues>

**LazyData** true

**RoxygenNote** 6.0.1

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Dewey Dunnington [aut, cre]

**Maintainer** Dewey Dunnington <dewey@fishandwhistle.net>

**Repository** CRAN

**Date/Publication** 2018-08-20 09:21:25

## R topics documented:

mudata2-package . . . . .	2
alta_lake . . . . .	3
as_mudata . . . . .	4
autoplot.mudata . . . . .	5

biplot.mudata . . . . .	6
collect.mudata . . . . .	6
distinct_params . . . . .	7
filter_datasets . . . . .	8
is_mudata . . . . .	9
kentvillegreenwood . . . . .	10
long_ggplot . . . . .	11
long_lake . . . . .	12
long_pairs . . . . .	12
mudata . . . . .	14
mudata_prepare_column . . . . .	15
mutate_data . . . . .	17
new_mudata . . . . .	18
ns_climate . . . . .	18
parallel_gather . . . . .	19
pocmaj . . . . .	20
pocmajsum . . . . .	21
print.mudata . . . . .	21
rbind.mudata . . . . .	22
rename_locations . . . . .	22
second_lake_temp . . . . .	24
select_datasets . . . . .	24
subset.mudata . . . . .	26
tbl_data . . . . .	27
update_columns_table . . . . .	28
update_datasets . . . . .	29
write_mudata . . . . .	30
<b>Index</b>	<b>32</b>

---

mudata2-package	<i>A (Mostly) Universal Data Format for Multi-Parameter, Spatiotemporal Data</i>
-----------------	--

---

## Description

The ‘mudata’ package for R is a set of tools to create, manipulate, and visualize multi-parameter, spatiotemporal data. Data of this type includes all data where multiple parameters (e.g. wind speed, precipitation, temperature) are measured along common axes (e.g. time, depth) at discrete locations (e.g. climate stations). These data include long-term climate data collected from climate stations, paleolimnological data, ice core data, long-term water quality monitoring data, and ocean core data among many others. Data of this type is often voluminous and difficult to organize/document given its multidimensional nature. The (mostly) universal data (mudata) format is an attempt to organize these data in a common way to facilitate their documentation and comparison.

## Details

The (mostly) universal data format is a collection of five (or more) tables, one of which contains the data in a parameter-long form (see [gather](#)). The easiest way to visualize a mudata object is to inspect the sample datasets within the package ([ns\\_climate](#), [kentvillegreenwood](#), [alta\\_lake](#), [long\\_lake](#), and [second\\_lake\\_temp](#)).

## Author(s)

**Maintainer:** Dewey Dunnington <dewey@fishandwhistle.net>

## References

Dunnington DW and Spooner IS (2018). "Using a linked table-based structure to encode self-describing multiparameter spatiotemporal data". FACETS. doi:10.1139/facets-2017-0026 <http://www.facetsjournal.com/doi/10.1139/facets-2017-0026>

## See Also

[mudata](#), [read\\_mudata](#)

## Examples

```
print(kentvillegreenwood)
autoplot(kentvillegreenwood)
```

---

alta\_lake

*Alta Lake Gravity Core Data*

---

## Description

Bulk geochemistry of a gravity core from Alta Lake, Whistler, British Columbia, Canada.

## Usage

```
alta_lake
```

## Format

A [mudata](#) object

## References

Dunnington DW, Spooner IS, White CE, et al (2016) A geochemical perspective on the impact of development at Alta Lake, British Columbia, Canada. J Paleolimnol 56:315-330. doi: 10.1007/s10933-016-9919-x

## Examples

```
library(ggplot2)
autoplot(alta_lake, y = "depth") + scale_y_reverse()
autoplot(alta_lake, y = "age")
```

---

as\_mudata

*Coerce objects to mudata*

---

## Description

Coerce objects to mudata

## Usage

```
as_mudata(x, ...)

as.mudata(x, ...)

## S3 method for class 'mudata'
as_mudata(x, ...)

## S3 method for class 'data.frame'
as_mudata(x, ...)

## S3 method for class 'tbl'
as_mudata(x, ...)

## S3 method for class 'src_sql'
as_mudata(x, ...)

## S3 method for class 'list'
as_mudata(x, ...)
```

## Arguments

x	An object
...	Passed to other methods

## Value

A [mudata](#) object or an error

---

autoplot.mudata	<i>Autoplot a mudata object</i>
-----------------	---------------------------------

---

## Description

Produces a quick graphical summary of a mudata object. The autoplot() function is based on ggplot2's [qplot](#), and is the preferred (and most flexible) plotting method. The plot() function uses base R graphics and produces quick summary plot, but is unlikely to be useful in any other context. Note that all column names must be quoted (i.e., aesthetic = "col\_name" not aesthetic = col\_name).

## Usage

```
## S3 method for class 'mudata'  
autoplot(object, facets = "param", col = "location",  
         pch = "dataset", ...)  
  
## S3 method for class 'mudata'  
plot(x, facets = "param", col = "location",  
     pch = "dataset", ...)
```

## Arguments

facets	Column to be used as facet column
col	Column to be used as colour aesthetic
pch	Column to be used as shape aesthetic
...	Passed on to <a href="#">long_plot</a> or <a href="#">long_ggplot</a>
x, object	A <a href="#">mudata</a> object

## See Also

[long\\_ggplot](#)

## Examples

```
# plot using base plot  
plot(kentvillegreenwood)  
  
# a more informative plot using ggplot  
autoplot(kentvillegreenwood)
```

---

biplot.mudata	<i>Biplot a mudata object</i>
---------------	-------------------------------

---

### Description

Uses [autobiplot](#) and [long\\_biplot](#) to produce parameter vs. parameter plots contained in a [mudata](#) object.

### Usage

```
## S3 method for class 'mudata'
biplot(x, ...)
```

```
## S3 method for class 'mudata'
autobiplot(x, ...)
```

### Arguments

x	A mudata object
...	passed to plotting methods

### Value

A [ggplot](#) object (autobiplot) or the result of [plot.default](#).

### Examples

```
kvtemp <- kentvillegreenwood %>% select_params(contains("temp"))

# use base plotting for regular biplot function
biplot(kvtemp)

# use ggplot and facet_grid to biplot
autobiplot(kvtemp, col = "location")
```

---

collect.mudata	<i>Collect all mudata components</i>
----------------	--------------------------------------

---

### Description

Objects created by [mudata](#) are generally assumed to be local data frames, but some methods may function on database tbls (especially in the future). This function applies [collect](#) to all component tables.

**Usage**

```
## S3 method for class 'mudata'
collect(x, ...)
```

**Arguments**

```
x          A mudata object
...        Passed to collect
```

**Value**

A mudata object with all components as local data frames.

---

distinct_params	<i>Get distinct params, locations, and datasets from a mudata object</i>
-----------------	--

---

**Description**

Get distinct params, locations, and datasets from a mudata object

**Usage**

```
distinct_params(x, ...)

## Default S3 method:
distinct_params(x, table = "data", ...)

distinct_locations(x, ...)

## Default S3 method:
distinct_locations(x, table = "data", ...)

distinct_datasets(x, ...)

## Default S3 method:
distinct_datasets(x, table = "data", ...)

distinct_columns(x, ...)

## Default S3 method:
distinct_columns(x, table = names(x), ...)

unique_params(x, table = "data")

unique_locations(x, table = "data")
```

```

unique_datasets(x, table = "data")

unique_columns(x, table = names(x))

## S3 method for class 'mudata'
src_tbls(x)

```

### Arguments

x	A mudata object
...	Passed to other methods
table	The table to use to calculate the distinct values. Using the "data" table is safest, but for large datasets that are not in memory, using the meta table (params, locations, or datasets) may be useful.

### Value

A character vector of distinct parameter names

### Examples

```

distinct_params(kentvillegreenwood)
distinct_locations(kentvillegreenwood)
distinct_datasets(kentvillegreenwood)

```

---

filter_datasets	<i>Subset a mudata object by complex expression</i>
-----------------	---

---

### Description

These methods allow more complex selection criteria than [select\\_datasets](#) and family, which only use the identifier values. These methods first subset the required table using the provided expression, then subset other tables to ensure internal consistency.

### Usage

```

filter_datasets(.data, ...)

## Default S3 method:
filter_datasets(.data, ...)

filter_data(.data, ...)

## Default S3 method:
filter_data(.data, ...)

filter_locations(.data, ...)

```



```
## Default S3 method:  
filter_locations(.data, ...)  
  
filter_params(.data, ...)  
  
## Default S3 method:  
filter_params(.data, ...)
```

### Arguments

.data	A <a href="#">mudata</a> object
...	Objects passed to <a href="#">filter</a> on the appropriate table

### Value

A subsetted mudata object

### See Also

[filter](#), [select\\_locations](#)

### Examples

```
# select only locations with a latitude above 45  
ns_climate %>%  
  filter_locations(latitude > 45)  
  
# select only params measured in mm  
ns_climate %>%  
  filter_params(unit == "mm")  
  
# select only june temperature from ns_climate  
library(lubridate)  
ns_climate %>%  
  filter_data(month(date) == 6)
```

---

is_mudata	<i>Test if an object is a mudata object</i>
-----------	---

---

### Description

Test if an object is a mudata object

### Usage

```
is_mudata(x)
```

```
is.mudata(x)
```

**Arguments**

x                    An object

**Value**

TRUE if the object is a mudata object, FALSE otherwise

**Examples**

```
is_mudata(kentvillegreenwood)
```

---

kentvillegreenwood     *Kentville/Greenwood Climate Data*

---

**Description**

Climate data for Kentville and Greenwood (Nova Scotia) for July and August of 1999.

**Usage**

```
kentvillegreenwood
```

**Format**

A `mudata` object

**Source**

Environment Canada via the 'rclimateca' package. <http://climate.weather.gc.ca/>

**Examples**

```
autoplot(kentvillegreenwood)
```

## Description

These functions are intended to quickly visualize plot a parameter-long data frame with a few variables that identify single rows in the value column. The function is optimised to plot data with a time axis data either horizontally (time on the x axis) or vertically (time on the y axis). Facets are intended to be by parameter, which is guessed based on the right-most discrete variable named in `id_vars`. In the context of a `mudata` object, this function almost always guesses the axes correctly, but these choices can be overridden.

## Usage

```
long_ggplot(.data, ..., max_facets = 9, facet_args = list())
```

```
long_plot(.data, id_vars = NULL, measure_var = "value", x = NULL,
  y = NULL, facets = NULL, geom = "path", error_var = NULL, ...,
  max_facets = 9, facet_args = list(), scales = list())
```

```
long_plot_base(.data, id_vars = NULL, measure_var = "value", x = NULL,
  y = NULL, facets = NULL, geom = "path", error_var = NULL, ...)
```

## Arguments

<code>.data</code>	A <code>data.frame</code>
<code>...</code>	Additional aesthetic mappings, passed on to (or treated similarly to) <a href="#">aes_string</a>
<code>max_facets</code>	Constrain the maximum number of facets, where available
<code>facet_args</code>	Passed on to <a href="#">facet_wrap</a>
<code>id_vars</code>	Columns that identify unique rows
<code>measure_var</code>	Column that contains values to be plotted
<code>x</code>	Column to be used on the x-axis
<code>y</code>	Column to be used on the y-axis
<code>facets</code>	Column(s) to be used as facetting variable (using <a href="#">facet_wrap</a> )
<code>geom</code>	Can be any combination of point, path, or line.
<code>error_var</code>	The column to be used for plus/minus error bars
<code>scales</code>	Customize aesthetic mapping in <code>long_plot()</code>

## Examples

```
library(tidyr)
library(dplyr)
pocmaj_long <- pocmajsum %>%
  select(core, depth, Ca, Ti, V) %>%
```

```
gather(Ca, Ti, V, key = "variable", value = "value")
long_plot(pocmaj_long, col="core")
long_ggplot(pocmaj_long, col="core")
```

---

long\_lake

*Long Lake Lake Gravity/Percussion Core Data*


---

### Description

Bulk geochemistry of a gravity core from Long Lake, Cumberland Marshes Region, Nova Scotia-New Brunswick Border Region, Canada.

### Usage

```
long_lake
```

### Format

A [mudata](#) object

### References

Dunnington DW, White H, Spooner IS, et al (2017) A paleolimnological archive of metal sequestration and release in the Cumberland Basin Marshes, Atlantic Canada. *FACETS* 2:440-460. doi: 10.1139/facets-2017-0004

### Examples

```
library(ggplot2)
autoplot(long_lake, y = "depth") + scale_y_reverse()
```

---

long\_pairs

*Biplot a parameter-long data frame*


---

### Description

Use either the ggplot framework (autoplot) or base plotting to biplot a parameter-long data frame, like that of the data table in a [mudata](#) object.

**Usage**

```

long_pairs(x, id_vars, name_var, names_x = NULL, names_y = NULL,
  validate = TRUE, max_names = 5)

long_biplot(x, id_vars, name_var, measure_var = "value", names_x = NULL,
  na.rm = FALSE, max_names = 5, ...)

autobiplot(x, ...)

## S3 method for class 'data.frame'
autobiplot(x, id_vars, name_var, measure_var = "value",
  names_x = NULL, names_y = NULL, error_var = NULL, na.rm = TRUE,
  validate = TRUE, max_names = 5, labeller = ggplot2::label_value, ...)

```

**Arguments**

x	the object to biplot
id_vars	the columns that identify a single row in x
name_var	The column where names_x and names_y are to be found
names_x	The names to be included in the x axes, or all the names to be included
names_y	The names to be included on the y axes, or NULL for all possible combinations of names_x.
validate	Ensure id_vars identify unique rows
max_names	When guessing which parameters to biplot/pair, use only the first max_names (or FALSE to use all names)
measure_var	The column containing the values to plot
na.rm	Should NA values in measure_var be removed?
...	passed to aes_string()
error_var	The column containing values for error bars (plus or minus error_var).
labeller	The labeller to use to label facets (may want to use label_parsed to use plotmath-style labels)

**Examples**

```

library(tidyr)
library(dplyr)

# create a long, summarised representation of pocmaj data
pocmaj_long <- pocmajsum %>%
  select(core, depth, Ca, Ti, V) %>%
  gather(Ca, Ti, V, key = "param", value = "value")

# biplot using base plotting
long_biplot(pocmaj_long, id_vars = c("core", "depth"), name_var = "param")

# biplot using ggplot

```

```

autobiplot(pocmaj_long, id_vars = c("core", "depth"), name_var = "param")

# get the raw data used
long_pairs(pocmaj_long, id_vars = c("core", "depth"), name_var = "param")

```

---

mudata

---

*Create a mudata object*


---

## Description

Create a mudata object, which is a collection of five tables: data, locations, params, datasets, and columns. You are only required to provide the data table, which must contain columns "param" and "value", but will more typically contain columns "location", "param", "datetime" (or "date"), and "value". See [ns\\_climate](#), [kentvillegreenwood](#), [alta\\_lake](#), [long\\_lake](#), and [second\\_lake\\_temp](#) for examples of data in this format.

## Usage

```

mudata(data, locations = NULL, params = NULL, datasets = NULL,
        columns = NULL, x_columns = NULL, ..., more_tbls = NULL,
        dataset_id = "default", location_id = "default", validate = TRUE)

```

## Arguments

data	A <code>data.frame</code> / <code>tibble</code> containing columns "param" and "value" (at least), but more typically columns "location", "param", "datetime" (or "date", depending on the type of data), and "value".
locations	The locations table, which is a data frame containing the columns (at least) "dataset", and "location". If omitted, it will be created automatically using all unique dataset/location combinations.
params	The params table, which is a data frame containing the columns (at least) "dataset", and "param". If omitted, it will be created automatically using all unique dataset/param combinations.
datasets	The datasets table, which is a data frame containing the column (at least) "dataset". If omitted, it will be generated automatically using all unique datasets.
columns	The columns table, which is a data frame containing the columns (at least) "dataset", "table", and "column". If omitted, it will be created automatically using all dataset/table/column combinations.
x_columns	A vector of column names from the data table that in combination with "dataset", "location", and "param" identify unique rows. These will typically be guessed using the column names between "param" and "value".
..., more_tbls	More tbls (as named arguments) to be included in the mudata object
dataset_id	The dataset to use if a "dataset" column is omitted.
location_id	The location if a "location" column is omitted.
validate	Pass FALSE to skip validation of input tables using <a href="#">validate_mudata</a> .

## Value

An object of class "mudata", which is a [list](#) with components data, locations, params, datasets, columns, and any other tables provided in more\_tbls. All list components must be tbls.

## References

Dunnington DW and Spooner IS (2018). "Using a linked table-based structure to encode self-describing multiparameter spatiotemporal data". FACETS. doi:10.1139/facets-2017-0026 <http://www.facetsjournal.com/doi/10.1139/facets-2017-0026>

## Examples

```
# use the data table from kentvillegreenwood as a template
kg_data <- tbl_data(kentvillegreenwood)
# create mudata object using just the data table
mudata(kg_data)

# create a mudata object starting from a parameter-wide data frame
library(tidyr)
library(dplyr)

# gather columns and summarise replicates
datatable <- pocmaj %>%
  gather(Ca, Ti, V, key = "param", value = "param_value") %>%
  group_by(core, param, depth) %>%
  summarise(value = mean(param_value), sd = mean(param_value)) %>%
  rename(location = core)

# create mudata object
mudata(datatable)
```

---

mudata\_prepare\_column *Prepare mudata table columns for writing*

---

## Description

This set of generics is similar to [output\\_column](#) in that it converts columns to a form suitable to writing. `mudata_prepare_column` in combination with `is_intended_to_be_opposites` with `mudata_parse_column` except for date/time vectors that are not in UTC (`mudata_parse_column` assumes UTC, and `mudata_prepare_column` always converts to UTC with a message).

## Usage

```
mudata_prepare_column(x, format = NA, ...)

mudata_prepare_tbl(x, format = NA, ...)
```

```

## Default S3 method:
mudata_prepare_tbl(x, format = NA, ...)

## S3 method for class 'tbl'
mudata_prepare_tbl(x, format = NA, ...)

## S3 method for class 'data.frame'
mudata_prepare_tbl(x, format = NA, ...)

## Default S3 method:
mudata_prepare_column(x, format = NA, ...)

## S3 method for class 'POSIXt'
mudata_prepare_column(x, format = NA, ...)

## S3 method for class 'sfc'
mudata_prepare_column(x, format = NA, ...)

## S3 method for class 'hms'
mudata_prepare_column(x, format = NA, ...)

## S3 method for class 'list'
mudata_prepare_column(x, format = NA, ...)

mudata_parse_column(x, type_str = NA_character_, ...)

mudata_parse_tbl(x, type_str = NA_character_, ...)

```

### Arguments

<code>x</code>	A an object
<code>format</code>	csv, json, or NA for unknown,
<code>...</code>	Passed to methods
<code>type_str</code>	A type string, generated by the internal <code>generate_type_str</code>

### Details

Type strings are currently internal, and are in the columns table in the "type" column. They are usually one of "character", "date", "datetime", "double", "integer", "json", and "wkt". They can also contain simple arguments, like "wkt(epsg=4326)" (actually, "wkt" is the only type string that should have arguments). You should generally not mess with these (in fact, the "type" column in the columns table is overwritten right before read by default, so it is hard to mess this up).

### Value

An atomic vector



---

mutate_data	<i>Modify mudata tables</i>
-------------	-----------------------------

---

## Description

Modify mudata tables

## Usage

```
mutate_data(x, ...)  
mutate_params(x, ...)  
mutate_locations(x, ...)  
mutate_datasets(x, ...)  
mutate_columns(x, ...)  
mutate_tbl(x, ...)  
  
## Default S3 method:  
mutate_tbl(x, tbl, ...)
```

## Arguments

x	A mudata object
...	Passed to <a href="#">mutate</a>
tbl	The table name to modify

## Value

A modified mudata object

## Examples

```
library(lubridate)  
second_lake_temp %>%  
  mutate_data(datetime = with_tz(datetime, "America/Halifax"))
```

---

new_mudata	<i>Validate, create a mudata object</i>
------------	---

---

### Description

Validates a mudata object by calling [stop](#) when an error is found; creates a mudata object from a [list](#). Validation is generally performed when objects are created using [mudata](#), or when objects are read/written using [read\\_mudata](#) and [write\\_mudata](#).

### Usage

```
new_mudata(md, x_columns)

validate_mudata(md, check_unique = TRUE, check_references = TRUE,
  action = stop)
```

### Arguments

md	An object of class 'mudata'
x_columns	The x_columns attribute (see <a href="#">mudata</a> ).
check_unique	Check if columns identify unique values in the appropriate tables
check_references	Check the referential integrity of the mudata object
action	The function to be called when errors are detected in validate_mudata

### Examples

```
validate_mudata(kentvillegreenwood)
new_mudata(kentvillegreenwood, x_columns = "date")
```

---

ns_climate	<i>Nova Scotia Long-Term Climate Data</i>
------------	---

---

### Description

Monthly climate data for locations in Nova Scotia with records longer than 80 years.

### Usage

```
ns_climate
```

### Format

A [mudata](#) object

**Source**

Environment Canada via the 'rclimateca' package. <http://climate.weather.gc.ca/>

**Examples**

```
print(ns_climate)
autoplot(ns_climate) # quite a messy plot, lots of data

# a more focused plot comparing three locations
library(lubridate)
ns_climate %>%
  select_locations(sable_island = starts_with("SABLE"),
                  nappan = starts_with("NAPPAN"),
                  baddeck = starts_with("BADDECK")) %>%
  select_params(ends_with("temp")) %>%
  filter_data(month(date) == 6) %>%
  autoplot()
```

---

parallel\_gather

*Melt multiple sets of columns in parallel*


---

**Description**

Essentially this is a wrapper around [gather](#) that is able to [bind\\_cols](#) with several gather operations. This is useful when a wide data frame contains uncertainty or flag information in paired columns.

**Usage**

```
parallel_gather(x, key, ..., convert = FALSE, factor_key = FALSE)
```

**Arguments**

x	A data.frame
key	Column name to use to store variables, which are the column names of the first gather operation.
...	Named arguments in the form <code>new_col_name = c(old, col, names)</code> . All named arguments must have the same length (i.e., gather the same number of columns).
convert	Convert types (see <a href="#">gather</a> )
factor_key	Control whether the key column is a factor or character vector.

**Value**

A gathered data frame.

**See Also**[gather](#)**Examples**

```
# gather paired value/error columns using
# parallel_gather
parallel_gather(pocmajsum, key = "param",
               value = c(Ca, Ti, V),
               sd = c(Ca_sd, Ti_sd, V_sd))

# identical result using only tidyverse functions
library(dplyr)
library(tidyr)
gathered_values <- pocmajsum %>%
  select(core, depth, Ca, Ti, V) %>%
  gather(Ca, Ti, V,
         key = "param", value = "value")
gathered_sds <- pocmajsum %>%
  select(core, depth, Ca_sd, Ti_sd, V_sd) %>%
  gather(Ca_sd, Ti_sd, V_sd,
         key = "param_sd", value = "sd")

bind_cols(
  gathered_values,
  gathered_sds %>% select(sd)
)
```

---

pocmaj

*Pockwock Lake/Lake Major Elemental Sample Data*

---

**Description**

A small example data.frame used to test structure methods.

**Usage**

```
pocmaj
```

**Format**

A data.frame containing multi-qualifier concentration data

---

pocmajsum	<i>Pre-summarised Sample Data</i>
-----------	-----------------------------------

---

**Description**

A small example data.frame of pre-summarised data; a summarised version of the [pocmaj](#) dataset.

**Usage**

```
pocmajsum
```

**Format**

A data.frame containing multi-qualifier data

---

print.mudata	<i>Print a mudata object</i>
--------------	------------------------------

---

**Description**

Print a mudata object

**Usage**

```
## S3 method for class 'mudata'
print(x, ..., width = NULL)
```

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

**Arguments**

x, object	A mudata object
...	Passed to other methods
width	The number of characters to use as console width

**Value**

print returns x (invisibly); summary returns a data frame with summary information.

**Examples**

```
print(kentvillegreenwood)
```

---

rbind.mudata	<i>Combine mudata objects</i>
--------------	-------------------------------

---

### Description

This implementation of `rbind` combines component tables using `bind_rows` and `distinct`. When combined object use different datasets, or when subsets of the same object are recombined, this function works well. When this is not the case, it may be necessary to modify the tables such that when they are passed to `bind_rows` and `distinct`, no duplicate information exists. This should be picked up by `validate_mudata`.

### Usage

```
## S3 method for class 'mudata'
rbind(..., validate = TRUE)
```

### Arguments

...                    `mudata` objects to combine  
 validate              Flag to validate the final object using `validate_mudata`.

### Value

A mudata object

### Examples

```
rbind(
  kentvillegreenwood %>%
    select_params(maxtemp) %>%
    select_locations(starts_with("KENT")),
  kentvillegreenwood %>%
    select_params(mintemp) %>%
    select_locations(starts_with("GREEN"))
)
```

---

rename_locations	<i>Rename identifiers in a mudata object</i>
------------------	--

---

### Description

These functions rename locations, datasets, params, and columns, making sure internal consistency is maintained. These functions use dplyr syntax for renaming (i.e. the `rename` function). This syntax can also be used while subsetting using `select_locations` and family.

**Usage**

```
rename_locations(.data, ...)  
  
## Default S3 method:  
rename_locations(.data, ...)  
  
rename_params(.data, ...)  
  
## Default S3 method:  
rename_params(.data, ...)  
  
rename_datasets(.data, ...)  
  
## Default S3 method:  
rename_datasets(.data, ...)  
  
rename_columns(.data, ...)  
  
## Default S3 method:  
rename_columns(.data, ...)
```

**Arguments**

<code>.data</code>	A mudata object
<code>...</code>	Variables to rename in the form <code>new_var = old_var</code>

**Value**

A modified mudata object

**See Also**

[rename](#), [select\\_locations](#)

**Examples**

```
rename_datasets(kentvillegreenwood, avalley = ecclimate)  
rename_locations(kentvillegreenwood, Greenwood = starts_with("GREENWOOD"))  
rename_params(kentvillegreenwood, max_temp = maxtemp)  
rename_columns(kentvillegreenwood, lon = longitude, lat = latitude)
```

---

second_lake_temp	<i>Second Lake Thermistor String Data</i>
------------------	---

---

### Description

Temperatures at multiple depths in the water column for a season at Second Lake, Lower Sackville, Nova Scotia, Canada.

### Usage

```
second_lake_temp
```

### Format

A [mudata](#) object

### References

Misiuk B (2014) A multi-proxy comparative paleolimnological study of anthropogenic impact between First and Second Lake, Lower Sackville, Nova Scotia. B.Sc.H. Thesis, Acadia University

### Examples

```
library(ggplot2)
autoplot(second_lake_temp, y = "depth", x = "datetime",
         col = "value", geom = "point") +
  scale_y_reverse()
autoplot(second_lake_temp, x = "datetime", y = "value",
         facets = c("param", "depth"))
```

---

select_datasets	<i>Subset a mudata object by identifier</i>
-----------------	---

---

### Description

These functions use dplyr-like selection syntax to quickly subset a mudata object by param, location, or dataset. Params, locations, and datasets can also be renamed using keyword arguments, identical to dplyr selection syntax.



**Usage**

```

select_datasets(.data, ...)

## Default S3 method:
select_datasets(.data, ..., .factor = FALSE)

select_locations(.data, ...)

## Default S3 method:
select_locations(.data, ..., .factor = FALSE)

select_params(.data, ...)

## Default S3 method:
select_params(.data, ..., .factor = FALSE)

```

**Arguments**

<code>.data</code>	A mudata object
<code>...</code>	Quoted names, bare names, or helpers like <a href="#">starts_with</a> , <a href="#">contains</a> , <a href="#">ends_with</a> , <a href="#">one_of</a> , or <a href="#">matches</a> .
<code>.factor</code>	If TRUE, the new object will keep the order specified by converting columns to factors. This may be useful for specifying order when using <a href="#">autoplot.mudata</a> .

**Value**

A subsetting mudata object.

**See Also**

[select](#), [rename\\_locations](#), [distinct\\_locations](#), [filter\\_locations](#)

**Examples**

```

# renaming can be handy when locations are verbosely named
ns_climate %>%
  select_locations(sable_island = starts_with("SABLE"),
                  nappan = starts_with("NAPPAN"),
                  baddeck = starts_with("BADDECK")) %>%
  select_params(ends_with("temp"))

# can also use quoted values
long_lake %>%
  select_params("Pb", "As", "Cr")

# can also use negative values to remove params/datasets/locations
long_lake %>%
  select_params(-Pb)

# to get around non-standard evaluation, use one_of()

```

```
my_params <- c("Pb", "As", "Cr")
long_lake %>%
  select_params(one_of(my_params))
```

---

subset.mudata

*Subset a MuData object*

---

### Description

This object uses standard evaluation to subset a [mudata](#) object using character vectors of datasets, params, and locations. The result is subsetting such that all rows in the data table are documented in the other tables (provided) they were to begin with. It is preferred to use [select\\_locations](#), [select\\_params](#), and [select\\_datasets](#) to subset a mudata object, or [filter\\_data](#), [filter\\_locations](#), [filter\\_params](#), and [filter\\_datasets](#) to subset by row while maintaining internal consistency.

### Usage

```
## S3 method for class 'mudata'
subset(x, ..., datasets = NULL, params = NULL,
       locations = NULL)
```

### Arguments

x	The object to subset
...	Used to <a href="#">filter</a> the data table
datasets	Vector of datasets to include
params	Vector of parameters to include
locations	Vector of locations to include

### Value

A subsetting mudata object

### See Also

[select\\_locations](#), [select\\_params](#), [select\\_datasets](#), [filter\\_data](#), [filter\\_locations](#), [filter\\_params](#), and [filter\\_datasets](#)

### Examples

```
subset(kentvillegreenwood, params = c("mintemp", "maxtemp"))
```

---

`tbl_data`*Access components of a mudata object*

---

**Description**

Access components of a mudata object

**Usage**

```
tbl_data(x)
```

```
## Default S3 method:
```

```
tbl_data(x)
```

```
tbl_data_wide(x, ...)
```

```
## Default S3 method:
```

```
tbl_data_wide(x, key = "param", value = "value", ...)
```

```
tbl_params(x)
```

```
## Default S3 method:
```

```
tbl_params(x)
```

```
tbl_locations(x)
```

```
## Default S3 method:
```

```
tbl_locations(x)
```

```
tbl_datasets(x)
```

```
## Default S3 method:
```

```
tbl_datasets(x)
```

```
tbl_columns(x)
```

```
tbl_columns(x)
```

```
## S3 method for class 'mudata'
```

```
tbl(src, which, ...)
```

```
x_columns(x)
```

```
## Default S3 method:
```

```
x_columns(x)
```

**Arguments**

x, src	A mudata object
...	Passed to other methods
key, value	Passed to <a href="#">spread</a>
which	Which tbl to extract

**Value**

The appropriate component

**Examples**

```
tbl_data(kentvillegreenwood)
```

---

update\_columns\_table *Update the columns table*

---

**Description**

Update the columns table

**Usage**

```
update_columns_table(md, quiet = FALSE)
```

**Arguments**

md	A mudata object
quiet	Suppress changes to existing types

**Value**

A mudata object

---

update\_datasets      *Add documentation to mudata objects*

---

## Description

Add documentation to mudata objects

## Usage

```
update_datasets(x, ...)  
  
## Default S3 method:  
update_datasets(x, datasets, ...)  
  
update_locations(x, ...)  
  
## Default S3 method:  
update_locations(x, locations, datasets, ...)  
  
update_params(x, ...)  
  
## Default S3 method:  
update_params(x, params, datasets, ...)  
  
update_columns(x, ...)  
  
## Default S3 method:  
update_columns(x, columns, tables, datasets, ...)
```

## Arguments

x	A mudata object
...	Key/value pairs (values of length 1)
datasets	One or more datasets to update
locations	One or more locations to update
params	One or more params to update
columns	One or more columns to update (columns table)
tables	One or more tables to update (columns table)

## Value

A modified version of x

**Examples**

```
kentvillegreenwood %>%
  update_datasets("ecclimate", new_key = "new_value") %>%
  tbl_datasets()
```

---

write_mudata	<i>Read/Write mudata objects</i>
--------------	----------------------------------

---

**Description**

These functions will read and write mudata objects to disk using a directory (which contains one .csv file for each table in the object), a ZIP archive (which is a zipped version of the directory format), or a JSON file. The base read/write functions attempt to guess which of these types to use based on the file extension: use the specific read/write function to avoid this.

**Usage**

```
write_mudata(md, filename, ...)
```

```
read_mudata(filename, ...)
```

```
write_mudata_zip(md, filename, overwrite = FALSE, validate = TRUE,
  update_columns = TRUE, ...)
```

```
read_mudata_zip(filename, validate = TRUE, ...)
```

```
write_mudata_dir(md, filename, overwrite = FALSE, validate = TRUE,
  update_columns = TRUE, ...)
```

```
read_mudata_dir(filename, validate = TRUE, ...)
```

```
write_mudata_json(md, filename, overwrite = FALSE, validate = TRUE,
  update_columns = TRUE, pretty = TRUE, ...)
```

```
to_mudata_json(md, validate = TRUE, update_columns = TRUE, pretty = FALSE,
  ...)
```

```
read_mudata_json(filename, validate = TRUE, ...)
```

```
from_mudata_json(txt, validate = TRUE, ...)
```

**Arguments**

md	A mudata object
filename	File to read/write (can also be a directory)
...	Passed to read/write functions

overwrite	Pass TRUE to overwrite if the file/directory already exists.
validate	Flag to validate mudata object after read or before write
update_columns	Update the columns table "type" column to reflect the internal R types of columns (recommened).
pretty	Produce pretty or minified JSON output
txt	JSON text from which to read a mudata object.

### Details

These functions are designed to make sure that the read/write operations are as lossless as possible. Some exceptions to this are if date/time columns are not in UTC (in which case they will be converted to UTC before writing), and if table names have characters that are not filesystem safe (allowed characters are [A-Za-z0-9\_.-] and others will be stripped).

### Examples

```
# read/write to directory
outfile <- tempfile(fileext=".mudata")
write_mudata(kentvillegreenwood, outfile)
md <- read_mudata(outfile)
unlink(outfile)

# read/write to zip
outfile <- tempfile(fileext=".zip")
write_mudata(kentvillegreenwood, outfile)
md <- read_mudata(outfile)
unlink(outfile)

# read/write to JSON
outfile <- tempfile(fileext=".json")
write_mudata(kentvillegreenwood, outfile)
md <- read_mudata(outfile)
unlink(outfile)
```

# Index

## \*Topic **datasets**

- alta\_lake, 3
  - kentvillegreenwood, 10
  - long\_lake, 12
  - ns\_climate, 18
  - pocmaj, 20
  - pocmajsum, 21
  - second\_lake\_temp, 24
- aes\_string, 11
- alta\_lake, 3, 3, 14
- as.mudata (as\_mudata), 4
- as\_mudata, 4
- autobiplot, 6
- autobiplot (long\_pairs), 12
- autobiplot.mudata (biplot.mudata), 6
- autoplot.mudata, 5, 25
- bind\_cols, 19
- bind\_rows, 22
- biplot.mudata, 6
- collect, 6, 7
- collect.mudata, 6
- contains, 25
- distinct, 22
- distinct\_columns (distinct\_params), 7
- distinct\_datasets (distinct\_params), 7
- distinct\_locations, 25
- distinct\_locations (distinct\_params), 7
- distinct\_params, 7
- ends\_with, 25
- facet\_wrap, 11
- filter, 9, 26
- filter\_data, 26
- filter\_data (filter\_datasets), 8
- filter\_datasets, 8, 26
- filter\_locations, 25, 26
- filter\_locations (filter\_datasets), 8
- filter\_params, 26
- filter\_params (filter\_datasets), 8
- from\_mudata\_json (write\_mudata), 30
- gather, 3, 19, 20
- ggplot, 6
- is.mudata (is\_mudata), 9
- is\_mudata, 9
- kentvillegreenwood, 3, 10, 14
- list, 15, 18
- long\_biplot, 6
- long\_biplot (long\_pairs), 12
- long\_ggplot, 5, 11
- long\_lake, 3, 12, 14
- long\_pairs, 12
- long\_plot, 5
- long\_plot (long\_ggplot), 11
- long\_plot\_base (long\_ggplot), 11
- matches, 25
- mudata, 3–6, 9–12, 14, 18, 22, 24, 26
- mudata2 (mudata2-package), 2
- mudata2-package, 2
- mudata\_parse\_column  
(mudata\_prepare\_column), 15
- mudata\_parse\_tbl  
(mudata\_prepare\_column), 15
- mudata\_prepare\_column, 15
- mudata\_prepare\_tbl  
(mudata\_prepare\_column), 15
- mutate, 17
- mutate\_columns (mutate\_data), 17
- mutate\_data, 17
- mutate\_datasets (mutate\_data), 17
- mutate\_locations (mutate\_data), 17
- mutate\_params (mutate\_data), 17
- mutate\_tbl (mutate\_data), 17



new\_mudata, 18  
ns\_climate, 3, 14, 18

one\_of, 25  
output\_column, 15

parallel\_gather, 19  
plot.default, 6  
plot.mudata (autoplot.mudata), 5  
pocmaj, 20, 21  
pocmajsum, 21  
print.mudata, 21

qplot, 5

rbind, 22  
rbind.mudata, 22  
read\_mudata, 3, 18  
read\_mudata (write\_mudata), 30  
read\_mudata\_dir (write\_mudata), 30  
read\_mudata\_json (write\_mudata), 30  
read\_mudata\_zip (write\_mudata), 30  
rename, 22, 23  
rename\_columns (rename\_locations), 22  
rename\_datasets (rename\_locations), 22  
rename\_locations, 22, 25  
rename\_params (rename\_locations), 22

second\_lake\_temp, 3, 14, 24  
select, 25  
select\_datasets, 8, 24, 26  
select\_locations, 9, 22, 23, 26  
select\_locations (select\_datasets), 24  
select\_params, 26  
select\_params (select\_datasets), 24  
spread, 28  
src\_tbls.mudata (distinct\_params), 7  
starts\_with, 25  
stop, 18  
subset.mudata, 26  
summary.mudata (print.mudata), 21

tbl.mudata (tbl\_data), 27  
tbl\_columns (tbl\_data), 27  
tbl\_data, 27  
tbl\_data\_wide (tbl\_data), 27  
tbl\_datasets (tbl\_data), 27  
tbl\_locations (tbl\_data), 27  
tbl\_params (tbl\_data), 27  
tibble, 14

to\_mudata\_json (write\_mudata), 30

unique\_columns (distinct\_params), 7  
unique\_datasets (distinct\_params), 7  
unique\_locations (distinct\_params), 7  
unique\_params (distinct\_params), 7  
update\_columns (update\_datasets), 29  
update\_columns\_table, 28  
update\_datasets, 29  
update\_locations (update\_datasets), 29  
update\_params (update\_datasets), 29

validate\_mudata, 14, 22  
validate\_mudata (new\_mudata), 18

write\_mudata, 18, 30  
write\_mudata\_dir (write\_mudata), 30  
write\_mudata\_json (write\_mudata), 30  
write\_mudata\_zip (write\_mudata), 30

x\_columns (tbl\_data), 27