

# Package ‘ofpetrial’

May 15, 2024

**Type** Package

**Title** Design on-Farm Precision Field Agronomic Trials

**Version** 0.1.1

**Maintainer** Taro Mieno <tmieno2@unl.edu>

**URL** <https://difm-brain.github.io/ofpetrial/>

**BugReports** <https://github.com/DIFM-Brain/ofpetrial/issues>

**Description** A comprehensive system for designing and implementing on-farm precision field agronomic trials. You provide field data, tell 'ofpetrial' how to design a trial, and get readily-usable trial design files and a report checks the validity and reliability of the trial design.

**License** GPL (>= 3)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**Imports** data.table, ggplot2, sf, lwgeom, purrr, tidyr, tibble, ggpubr, ggExtra, terra, zip, rmarkdown, tmap, magrittr, dplyr, bookdown, leaflet

**VignetteBuilder** knitr

**Depends** R (>= 2.10)

**Suggests** knitr

**NeedsCompilation** no

**Author** Taro Mieno [aut, cre, cph] (<<https://orcid.org/0000-0002-0614-0771>>),  
Brittani Edge [aut, ctb]

**Repository** CRAN

**Date/Publication** 2024-05-15 08:50:03 UTC

## R topics documented:

add_blocks	2
assign_rates	3

change_rates . . . . .	4
check_alignment . . . . .	5
check_ortho_inputs . . . . .	5
check_ortho_with_chars . . . . .	6
exp_data . . . . .	7
make_exp_plots . . . . .	8
make_trial_report . . . . .	9
plot_info . . . . .	10
prep_plot . . . . .	10
prep_rate . . . . .	12
rate_info . . . . .	13
td_curved . . . . .	14
td_single_input . . . . .	15
td_two_input . . . . .	15
viz . . . . .	16
write_trial_files . . . . .	17

**Index** **18**

---

add_blocks	<i>Add blocks to trial design</i>
------------	-----------------------------------

---

**Description**

Delineate blocks on a trial design and assign block id to all the plots

**Usage**

```
add_blocks(td)
```

**Arguments**

td	trial design made by applying assign_rates() to experimental plots made by make_exp_plots()
----	---

**Value**

trial design with block\_id added

**Examples**

```
#--- load rate information ---#
data(td_single_input)

#--- add blocks ---#
td_with_blocks <- add_blocks(td_single_input)

#--- take a look ---#
td_with_blocks$trial_design
```

```
#--- visualize ---#  
viz(td_with_blocks, type = "block_id")
```

---

**assign\_rates***Assign rates to the plots of experimental plots*

---

### Description

This functions assign input rates for the plots created by `make_exp_plots()` according to the rate designs specified by the user in `rate_info`, which can be created by `prep_rateingle()`.

### Usage

```
assign_rates(exp_data, rate_info)
```

### Arguments

<code>exp_data</code>	experiment plots created by <code>make_exp_plots()</code>
<code>rate_info</code>	rate information created by <code>prep_rate()</code>

### Value

trial design as sf (experiment plots with rates assigned)

### Examples

```
#--- load experiment plots made by make_exp_plots() ---#  
data(exp_data)  
exp_data  
  
#--- load rate information ---#  
data(rate_info)  
rate_info  
  
#--- assign rates ---#  
td <- assign_rates(exp_data, rate_info)  
  
#--- visualization of the assigned rates ---#  
viz(td)
```

---

change_rates	<i>Change the assigned rates</i>
--------------	----------------------------------

---

**Description**

Change the assigned rates by plot and strip

**Usage**

```
change_rates(  
  td,  
  input_name = NA,  
  strip_ids,  
  plot_ids = NULL,  
  new_rates,  
  rate_by = "all"  
)
```

**Arguments**

td	trial design
input_name	(character) input name
strip_ids	(numeric) vector of strip_ids
plot_ids	(numeric) vector of plot_ids
new_rates	(numeric) single numeric number for 'rate_by = "all"', a vector of numeric values for 'rate_by = "strip"', a matrix of numeric numbers for 'rate_by = "plot"'. .
rate_by	(character) default is "all". The other options are "plot" and "strip".

**Value**

trial design with changed rates

**Examples**

```
### load rate information ###  
data(td_single_input)  
  
### change rates of some strips ###  
strip_ids <- 1:5  
plot_ids <- 5:10  
new_rates <- 200  
  
td_modified <- change_rates(td_single_input, "NH3", strip_ids, plot_ids, new_rates)  
  
### visualize ###  
viz(td_modified)
```

---

check_alignment	<i>Check the alignment of harvester and applicator/planter</i>
-----------------	--

---

**Description**

Check the alignment of harvester and applicator/planter for mixed treatment problems where multiple input rates are associated with yield monitor data

**Usage**

```
check_alignment(td)
```

**Arguments**

td                    trial design data created by make\_exp\_plots() and assign\_rates()

**Value**

a tibble

**Examples**

```
#--- load trial design ---#
data(td_single_input)

#--- check the alignment of harvester and applicator/planter ---#
machine_alignment <- check_alignment(td_single_input)

#--- check the degree of mixed treatment problem ---#
machine_alignment$overlap_data

#--- visualize the degree of mixed treatment problem ---#
machine_alignment$g_overlap[[1]]
```

---

check_ortho_inputs	<i>Check the correlation of the two inputs</i>
--------------------	--

---

**Description**

Check the correlation between the rates of the two inputs for a two-input experiment.

**Usage**

```
check_ortho_inputs(td)
```

**Arguments**

td                    trial design for a two-input experiment with rates assigned

**Value**

table

**Examples**

```
#--- load a trial design for a two-input experiment ---#
data(td_two_input)

#--- check correlation ---#
check_ortho_inputs(td_two_input)
```

---

check\_ortho\_with\_chars

*Check the orthogonality with field/topographic characteristics*

---

**Description**

Check the orthogonality of the trial input rates and observed characteristics provided by the user

**Usage**

```
check_ortho_with_chars(td, sp_data_list, vars_list)
```

**Arguments**

td	(tibble) trial design data created by <code>make_exp_plots()</code> and <code>assign_rates()</code>
sp_data_list	(list) list of spatial datasets as 'sf' from the 'sf' package or 'SpatRaster' from the 'terra' package
vars_list	(list) list of character vectors indicating the name of the variables to be used in the datasets specified in <code>sp_data_list</code>

**Value**

a list

**Examples**

```
data(td_single_input)
yield_sf <- sf::st_read(system.file("extdata", "yield-simple1.shp", package = "ofpetrial"))
ssurgo_sf <-
  sf::st_read(system.file("extdata", "ssurgo-simple1.shp", package = "ofpetrial")) %>%
  dplyr::mutate(mukey = factor(mukey))
topo_rast <-
  c(
    terra::rast(system.file("extdata", "slope.tif", package = "ofpetrial")),
    terra::rast(system.file("extdata", "twi.tif", package = "ofpetrial"))
  )
```

```

checks <-
  check_ortho_with_chars(
    td = td_single_input,
    sp_data_list = list(yield_sf, ssurgo_sf, topo_rast),
    vars_list = list("Yld_Vol_Dr", c("mukey", "clay"), names(topo_rast))
  )

checks$summary_data[[1]]

checks$summary_fig[[1]]

```

---

exp\_data

*Experiment data*


---

## Description

Data on the experiment created by running the ‘make\_exp\_plot()’ function, which includes various sf objects (e.g., experiment plots, ab-line, headland, etc). This data exists only for the purpose of making examples in some function references succinct.

## Usage

```
exp_data
```

## Format

tbl\_df tbl data.frame ‘exp\_data’ A data frame with 1 rows and 9 columns:

**input\_name** input name

**harvester\_width** width of the harvester

**plot\_width** width of the plots to be made

**field\_sf** field boundary as an sf object

**headland** headland as an sf object

**exp\_plots** experiment plots as an sf object

**ab\_lines** ab-lines for the applicator/planter as an sf object

**harvest\_ab\_lines** ab-lines for the harvester as an sf object

**abline\_type** (character) one of "free", "lock", "none" indicating the way ab-line is (or not) created

---

 make\_exp\_plots

*Make experimental plots/strips inside the field boundary*


---

### Description

Make experimental plots/strips inside the field boundary, harvester ab-line, and applicator/planter ab-line.

### Usage

```
make_exp_plots(
  input_plot_info,
  boundary_data,
  abline_data = NA,
  abline_type = "free"
)
```

### Arguments

`input_plot_info` (data.frame or a list of two data.frames) list of plot information created by `make_input_plot()`

`boundary_data` (character) path of the field boundary file or boundary as an sf

`abline_data` (character or sf) path of the ab-line file or ab-line as an sf

`abline_type` (character) the type of ab-line generation. Select from "free", "lock", and "none"

### Value

a tibble that include experimental plots as sf

### Examples

```
n_plot_info <-
  prep_plot(
    input_name = "NH3",
    unit_system = "imperial",
    machine_width = 30,
    section_num = 1,
    harvester_width = 20,
    headland_length = 30,
    side_length = 60
  )

exp_data <-
  make_exp_plots(
    input_plot_info = n_plot_info,
    boundary_data = system.file("extdata", "boundary-simple1.shp", package = "ofpetrial"),
    abline_data = system.file("extdata", "ab-line-simple1.shp", package = "ofpetrial"),
    abline_type = "free"
```



```
)  
exp_data$exp_plots
```

---

make\_trial\_report      *Create trial design report*

---

## Description

This function creates an html report describing the trial design created by the user with `assign_rates()` and includes figures showing machine alignment

## Usage

```
make_trial_report(td, folder_path, trial_name = NA, keep_rmd = FALSE)
```

## Arguments

<code>td</code>	trial design created by <code>assign_rates()</code>
<code>folder_path</code>	(character) path to the folder in which the report will be saved
<code>trial_name</code>	(character) name of trial to be used in report
<code>keep_rmd</code>	(logical) If FALSE (Default), the original rmd file will be deleted upon creating an html report. Otherwise, the rmd file will be saved in the folder specified by 'folder_path'.

## Value

path to the resulting html file (invisible)

## Examples

```
#--- load experiment made by assign_rates() ---#  
  
data(td_single_input)  
make_trial_report(  
  td = td_single_input,  
  folder_path = tempdir()  
)
```

---

plot_info	<i>Plot information</i>
-----------	-------------------------

---

### Description

Plot information for creating experiment plots using ‘make\_exp\_plot()’. This data exists only for the purpose of making examples in some function references succinct.

### Usage

```
plot_info
```

### Format

data.frame ‘plot\_info’ A data frame with 1 rows and 10 columns:

**input\_name** input name

**unit\_system** measurement system (metric or imperial)

**machine\_width** width of the applicator/planter

**section\_num** number of the sections of the machine

**section\_width** width of a section of the machine

**harvester\_width** width of the harvester

**plot\_width** width of the plots to be made

**headland\_length** length of the headland

**side\_length** length of the side

**min\_plot\_length** minimum plot length allowed

**max\_plot\_length** maximum plot length allowed

---

prep_plot	<i>Prepare plot information for a single-input experiment (length in meter)</i>
-----------	---

---

### Description

Prepare plot information for a single-input experiment case. All the length values need to be specified in meter.

**Usage**

```
prep_plot(
  input_name,
  unit_system,
  machine_width,
  section_num,
  harvester_width,
  plot_width = NA,
  headland_length = NA,
  side_length = NA,
  max_plot_width = NA,
  min_plot_length = NA,
  max_plot_length = NA
)
```

**Arguments**

input_name	(character) Input name
unit_system	(character) A character of either 'metric' or 'imperial' indicating the system of measurement used
machine_width	(numeric) A numeric number in units specified in unit_system that indicates the width of the applicator or planter of the input
section_num	(numeric) A numeric number that indicates the number of sections of the applicator or planter of the input
harvester_width	(numeric) A numeric number that indicates the width of the harvester
plot_width	(numeric) Default is c(NA, NA).
headland_length	(numeric) A numeric number that indicates the length of the headland (how long the non-experimental space is in the direction machines drive). Default is NA.
side_length	(numeric) A numeric number that indicates the length of the two sides of the field (how long the non-experimental space is in the direction perpendicular to the direction of machines). Default is NA.
max_plot_width	(numeric) Maximum width of the plots. Default is 36.576 meter (120 feet).
min_plot_length	(numeric) Minimum length of the plots. Default is 73.152 meter (240 feet).
max_plot_length	(numeric) Maximum length of the plots. Default is 91.440 meter (300 feet)

**Value**

a tibble with plot information necessary to create experiment plots

**Examples**

```

input_name <- "seed"
unit_system <- "metric"
machine_width <- 12
section_num <- 12
plot_width <- NA
harvester_width <- 24
prep_plot(input_name, unit_system, machine_width, section_num, harvester_width)

```

---

prep\_rate

---

*Create data of input rate information for a single input*


---

**Description**

Create data of input rate information for a single input with some checks on the validity of the information provided by the user. This can be used to assign rates to experiment plots using `assign_rates()`.

**Usage**

```

prep_rate(
  plot_info,
  gc_rate,
  unit,
  rates = NULL,
  min_rate = NA,
  max_rate = NA,
  num_rates = 5,
  design_type = NA,
  rank_seq_ws = NULL,
  rank_seq_as = NULL,
  rate_jump_threshold = NA
)

```

**Arguments**

plot_info	(data.frame) plot information created by <code>make_input_plot_data</code>
gc_rate	(numeric) Input rate the grower would have chosen if not running an experiment. This rate is assigned to the non-experiment part of the field. This rate also becomes one of the trial input rates unless you specify the trial rates directly using <code>rates</code> argument
unit	(string) unit of input
rates	(numeric vector) Default is <code>NULL</code> . Sequence of trial rates in the ascending order.
min_rate	(numeric) minimum input rate. Ignored if <code>rates</code> are specified.
max_rate	(numeric) maximum input rate. Ignored if <code>rates</code> are specified

num_rates	(numeric) Default is 5. It has to be an even number if design_type is "ejca". Ignored if rates are specified.
design_type	(string) type of trial design. available options are Latin Square ("ls"), Strip ("str"), Randomized Strip ("rstr"), Randomized Block ("rb"), Sparse ("sparse"), and Extra Jump-conscious Alternate "ejca". See the article on trial design for more details.
rank_seq_ws	(integer) vector of integers indicating the order of the ranking of the rates, which will be repeated "within" a strip.
rank_seq_as	(integer) vector of integers indicating the order of the ranking of the rates, which will be repeated "across" strip for their first plots.
rate_jump_threshold	(integer) highest jump in rate rank acceptable

**Value**

data.frame of input rate information

**Examples**

```
plot_info <-
  prep_plot(
    input_name = "seed",
    unit_system = "imperial",
    machine_width = 60,
    section_num = 24,
    harvester_width = 30,
    plot_width = 30
  )

prep_rate(
  plot_info,
  gc_rate = 30000,
  unit = "seeds",
  rates = c(20000, 25000, 30000, 35000, 40000)
)
```

---

rate\_info

*Rate information*

---

**Description**

Rate information for assigning rates to the experiment plots using the 'assign\_rates()' function. This data exists only for the purpose of making examples in some function references succinct.

**Usage**

rate\_info

**Format**

data.frame 'rate\_info' A data frame with 1 rows and 7 columns:

**input\_name** input name

**design\_type** type of the trial design to be created

**gc\_rate** normal rate the grower would have used if not running an experiment

**unit** unit of the input

**rates\_data** data.frame of rates and their ranks

**rank\_seq\_ws** vector of the ranking of rates that will repeated within a strip

**rank\_seq\_as** vector of the ranking of rates that will repeated as the first rate of the strips

---

 td\_curved | *Trial design (single-input) for a curved field* |**Description**

Trial design data created by assigning rates to experiment plots running the 'assign\_rates()' function. This data exists only for the purpose of making examples in some function references succinct.

**Usage**

```
td_curved
```

**Format**

tbl\_df tbl data.frame 'td\_curved' A data frame with 1 rows and 9 columns:

**input\_name** input name

**input\_type** shorthand for the type of the input: "N" for nitrogen, "S" for seed, etc.

**trial\_design** experiment plots with input rats assigned as an sf object

**design\_type** type of the trial design used

**unit** unit of the input

**abline\_type** (character) one of "free", "lock", "none" indicating the way ab-line is (or not) created

**ab\_lines** ab-lines for the applicator/planter as an sf object

**harvest\_ab\_lines** ab-lines for the harvester as an sf object

**field\_sf** field boundary as an sf object

**harvest\_width** width of the harvester

---

td_single_input	<i>Trial design (single-input)</i>
-----------------	------------------------------------

---

### Description

Trial design data created by assigning rates to experiment plots running the ‘assign\_rates()’ function. This data exists only for the purpose of making examples in some function references succinct.

### Usage

```
td_single_input
```

### Format

tbl\_df tbl data.frame ‘td\_single\_input’ A data frame with 1 rows and 9 columns:

**input\_name** input name

**input\_type** shorthand for the type of the input: "N" for nitrogen, "S" for seed, etc.

**trial\_design** experiment plots with input rats assigned as an sf object

**design\_type** type of the trial design used

**unit** unit of the input

**abline\_type** (character) one of "free", "lock", "none" indicating the way ab-line is (or not) created

**ab\_lines** ab-lines for the applicator/planter as an sf object

**harvest\_ab\_lines** ab-lines for the harvester as an sf object

**field\_sf** field boundary as an sf object

**harvest\_width** width of the harvester

---

td_two_input	<i>Trial design (two-input)</i>
--------------	---------------------------------

---

### Description

Trial design data created by assigning rates to experiment plots running the ‘assign\_rates()’ function. This data exists only for the purpose of making examples in some function references succinct.

### Usage

```
td_two_input
```

**Format**

`tbl_df` tbl data.frame 'td\_two\_input' A data frame with 1 rows and 9 columns:

**input\_name** input name

**input\_type** shorthand for the type of the input: "N" for nitrogen, "S" for seed, etc.

**trial\_design** experiment plots with input rats assigned as an sf object

**design\_type** type of the trial design used

**unit** unit of the input

**abline\_type** (character) one of "free", "lock", "none" indicating the way ab-line is (or not) created

**ab\_lines** ab-lines for the applicator/planter as an sf object

**harvest\_ab\_lines** ab-lines for the harvester as an sf object

**field\_sf** field boundary as an sf object

**harvest\_width** width of the harvester

---

viz

*Visualize various aspects of a trial design*

---

**Description**

Create plots of experiment rates, plot layout, `plot_id`, `strip_id`, and `block_id`, which can be specified by the 'type' argument.

**Usage**

```
viz(
  td,
  type = "rates",
  input_index = c(1, 2),
  text_size = 3,
  abline = FALSE,
  leaflet = FALSE
)
```

**Arguments**

<code>td</code>	(tibble) experiment plots made by <code>make_exp_plots()</code>
<code>type</code>	(character) type of plots to create. Available options are "rates", "layout", "plot_id", "strip_id", "block_id", "ab_line"
<code>input_index</code>	(numeric) a vector of length 1 or 2. 1 means the 1st input of the td, 2 means the second input of the td, and c(1, 2) means both of the inputs, which is the DEFAULT
<code>text_size</code>	(numeric) the size of plot ID, strip ID, and block ID numbers printed in the plots
<code>abline</code>	(logical) If TRUE, ab-lines are displayed as well. Default = FALSE. This applies only to type = "rates" and type = "layout".
<code>leaflet</code>	(logical) If TRUE, the plot will be superimposed on a satellite imagery of the field. Default is FALSE. This option is effective only for type = "rates".



**Value**

ggplot or leaflet (if leaflet == TRUE) object

**Examples**

```
#--- load trial design ---#
data(td_two_input)
viz(td_two_input)
```

---

write_trial_files	<i>Write trial design files for field implementation</i>
-------------------	--

---

**Description**

Write out all the necessary files to implement the trial design created. Exported files include

**Usage**

```
write_trial_files(td, folder_path, ext = "shp", zip = FALSE, zip_name = NA)
```

**Arguments**

td	(tibble) a tibble of a trial design created by applying assign_rate() to experimental plots made by make_exp_plots().
folder_path	(character) path to the folder in which the files will be saved
ext	(character) Default = "shp". Extension to use to save the files, "geojson" or any other extension supported by sf::st_write()
zip	(logical) Default = FALSE. If TRUE, all the files that are being written will be zipped.
zip_name	(character) name of the zip file created when zip = TRUE.

**Value**

nothing

**Examples**

```
#--- load trial design ---#
data(td_two_input)

write_trial_files(
  td = td_two_input,
  folder_path = tempdir(),
  zip = FALSE
)
```

# Index

## \* datasets

- exp\_data, [7](#)
- plot\_info, [10](#)
- rate\_info, [13](#)
- td\_curved, [14](#)
- td\_single\_input, [15](#)
- td\_two\_input, [15](#)

- add\_blocks, [2](#)
- assign\_rates, [3](#)

- change\_rates, [4](#)
- check\_alignment, [5](#)
- check\_ortho\_inputs, [5](#)
- check\_ortho\_with\_chars, [6](#)

- exp\_data, [7](#)

- make\_exp\_plots, [8](#)
- make\_trial\_report, [9](#)

- plot\_info, [10](#)
- prep\_plot, [10](#)
- prep\_rate, [12](#)

- rate\_info, [13](#)

- td\_curved, [14](#)
- td\_single\_input, [15](#)
- td\_two\_input, [15](#)

- viz, [16](#)

- write\_trial\_files, [17](#)