Package ‘FielDHub’

March 28, 2024

Title  A Shiny App for Design of Experiments in Life Sciences

Version  1.3.7

Description  A shiny design of experiments (DOE) app that aids in the creation of traditional,
un-replicated, augmented and partially-replicated designs applied to agriculture,
plant breeding, forestry, animal and biological sciences.

Depends  R (>= 4.1.0)

License  MIT + file LICENSE

Imports  config, golem, shiny (>= 1.7.0), htmltools, DT, shinythemes,
dplyr, numbers, blockdesign, shinycssloaders, ggplot2, plotly,
viridis, shinyalert, desplot, shinyjs

Encoding  UTF-8

RoxygenNote  7.2.3

Author  Didier Murillo [cre, aut],
        Salvador Gezan [aut],
        Ana Heilman [ctb],
        Thomas Walk [ctb],
        Johan Aparicio [ctb],
        Matthew Seefeldt [ctb],
        Jean-Marc Montpetit [ctb],
        Richard Horsley [ctb],
        North Dakota State University [cph]

Maintainer  Didier Murillo <didier.murilloflorez@ndsu.edu>

Suggests  testthat (>= 3.0.0), spelling, rlang, glue, knitr,
kableExtra, rmarkdown

Config/testthat/edition  3

URL  https://github.com/DidierMurilloF/FielDHub,
     https://didiermurillof.github.io/FielDHub/

BugReports  https://github.com/DidierMurilloF/FielDHub/issues

Language  en-US

VignetteBuilder  knitr
**NeedsCompilation**: no  
**Repository**: CRAN  
**Date/Publication**: 2024-03-28 05:40:02 UTC

### R topics documented:

- **alpha_lattice** ............................................. 2  
- **CRD** .......................................................... 4  
- **diagonal_arrangement** .................................... 6  
- **do_optim** ..................................................... 9  
- **full_factorial** ............................................. 11  
- **incomplete_blocks** ....................................... 13  
- **latin_square** ................................................ 14  
- **multi_location_prep** ..................................... 16  
- **optimized_arrangement** .................................. 18  
- **partially_replicated** ...................................... 21  
- **plot.FielDHub** ............................................. 23  
- **print.FielDHub** ........................................... 24  
- **print.fieldLayout** ......................................... 25  
- **print.summary.FielDHub** .................................. 26  
- **RCBD** ......................................................... 26  
- **RCBD_augmented** .......................................... 28  
- **rectangular_lattice** .................................... 30  
- **row_column** ................................................. 32  
- **run_app** ...................................................... 34  
- **sparse_allocation** ....................................... 34  
- **split_families** ............................................ 36  
- **split_plot** .................................................. 37  
- **split_split_plot** .......................................... 39  
- **square_lattice** ........................................... 41  
- **strip_plot** .................................................. 42  
- **summary.FielDHub** ........................................ 44  
- **swap_pairs** .................................................. 45

**Index** 47

---

**alpha_lattice**  
*Generates an Alpha Design*

**Description**

Randomly generates an alpha design like $\alpha(0, 1)$ across multiple locations.
Usage

alpha_lattice(
  t = NULL,
  k = NULL,
  r = NULL,
  l = 1,
  plotNumber = 101,
  locationNames = NULL,
  seed = NULL,
  data = NULL
)

Arguments

t  Number of treatments.

k  Size of incomplete blocks (number of units per incomplete block).

r  Number of full blocks (or resolvable replicates) (also number of replicates per treatment).

l  Number of locations. By default l = 1.

plotNumber  Numeric vector with the starting plot number for each location. By default plotNumber = 101.

locationNames  (optional) String with names for each of the l locations.

seed  (optional) Real number that specifies the starting seed to obtain reproducible designs.

data  (optional) Data frame with label list of treatments.

Value

A list with two elements.

- infoDesign is a list with information on the design parameters.
- fieldBook is a data frame with the alpha design field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References

Examples

# Example 1: Generates an alpha design with 4 full blocks and 15 treatments.
# Size of IBlocks k = 3.
alphalattice1 <- alpha_lattice(t = 15,
   k = 3,
   r = 4,
   l = 1,
   plotNumber = 101,
   locationNames = "GreenHouse",
   seed = 1247)

alphalattice1$infoDesign
head(alphalattice1$fieldBook, 10)

# Example 2: Generates an alpha design with 3 full blocks and 25 treatment.
# Size of IBlocks k = 5.
# In this case, we show how to use the option data.
treatments <- paste("G-", 1:25, sep = "")
ENTRY <- 1:25
treatment_list <- data.frame(list(ENTRY = ENTRY, TREATMENT = treatments))
head(treatment_list)
alphalattice2 <- alpha_lattice(t = 25,
   k = 5,
   r = 3,
   l = 1,
   plotNumber = 1001,
   locationNames = "A",
   seed = 1945,
   data = treatment_list)

alphalattice2$infoDesign
head(alphalattice2$fieldBook, 10)

---

CRD  Generates a Completely Randomized Design (CRD)

Description

It randomly generates a completely randomized design.

Usage

CRD(
   t = NULL,
   reps = NULL,
   plotNumber = 101,
   locationName = NULL,
   seed = NULL,
   data = NULL
)
Arguments

- **t**: An integer number with total number of treatments or a vector of dimension t with labels.
- **reps**: Number of replicates of each treatment.
- **plotNumber**: Starting plot number. By default `plotNumber = 101`.
- **locationName**: (optional) Name of the location.
- **seed**: (optional) Real number that specifies the starting seed to obtain reproducible designs.
- **data**: (optional) Data frame with the 2 columns with labels of each treatments and its number of replicates.

Value

A list with two elements.

- `infoDesign` is a list with information on the design parameters.
- `fieldBook` is a data frame with the CRD field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References


Examples

# Example 1: Generates a CRD design with 10 treatments and 5 reps each.
crd1 <- CRD(
  t = 10,
  reps = 5,
  plotNumber = 101,
  seed = 1987,
  locationName = "Fargo"
)
crd1$infoDesign
head(crd1$fieldBook, 10)

# Example 2: Generates a CRD design with 15 treatments and 6 reps each.
Gens <- paste("Wheat", 1:15, sep = "")
crd2 <- CRD(
  t = Gens,
  reps = 6,
  plotNumber = 1001,
  seed = 1654,
  locationName = "Fargo"
# Example 3: Generates a CRD design with 12 treatments and 4 reps each.
# In this case, we show how to use the option data.

treatments <- paste("ND-", 1:12, sep = "")
treatment_list <- data.frame(list(TREATMENT = treatments, REP = 4))
head(treatment_list)

crd3 <- CRD(
  t = NULL,
  reps = NULL,
  plotNumber = 2001,
  seed = 1655,
  locationName = "Cali",
  data = treatment_list
)
crd3$infoDesign
head(crd3$fieldBook, 10)

---

### diagonal_arrangement

**Spatial Un-replicated Diagonal Arrangement Design**

#### Description

Randomly generates an spatial un-replicated diagonal arrangement design.

#### Usage

```r
diagonal_arrangement(
  nrows = NULL,
  ncols = NULL,
  lines = NULL,
  checks = NULL,
  planter = "serpentine",
  l = 1,
  plotNumber = 101,
  kindExpt = "SUDC",
  splitBy = "row",
  seed = NULL,
  blocks = NULL,
  exptName = NULL,
  locationNames = NULL,
  multiLocationData = FALSE,
  data = NULL
)
```
diagonal_arrangement

Arguments

- **nrows**
  Number of rows in the field.

- **ncols**
  Number of columns in the field.

- **lines**
  Number of genotypes, experimental lines or treatments.

- **checks**
  Number of genotypes checks.

- **planter**
  Option for serpentine or cartesian plot arrangement. By default `planter = 'serpentine'`.

- **l**
  Number of locations or sites. By default `l = 1`.

- **plotNumber**
  Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.

- **kindExpt**
  Type of diagonal design, with single options: Single Un-replicated Diagonal Checks 'SUDC' and Decision Blocks Un-replicated Design with Diagonal Checks 'DBUDC' for multiple experiments. By default `kindExpt = 'SUDC'`.

- **splitBy**
  Option to split the field when `kindExpt = 'DBUDC'` is selected. By default `splitBy = 'row'`.

- **seed**
  (optional) Real number that specifies the starting seed to obtain reproducible designs.

- **blocks**
  Number of experiments or blocks to generate an `DBUDC` design. If `kindExpt = 'DBUDC'` and data is null, `blocks` are mandatory.

- **exptName**
  (optional) Name of the experiment.

- **locationNames**
  (optional) Names each location.

- **multiLocationData**
  (optional) Option to pass an entry list for multiple locations. By default `multiLocationData = FALSE`.

- **data**
  (optional) Data frame with 2 columns: ENTRY | NAME.

Value

A list with five elements.

- **infoDesign** is a list with information on the design parameters.
- **layoutRandom** is a matrix with the randomization layout.
- **plotsNumber** is a matrix with the layout plot number.
- **data_entry** is a data frame with the data input.
- **fieldBook** is a data frame with field book design. This includes the index (Row, Column).

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]
References


Examples

# Example 1: Generates a spatial single diagonal arrangement design in one location
# with 270 treatments and 30 check plots for a field with dimensions 15 rows x 20 cols
# in a serpentine arrangement.
spatd <- diagonal_arrangement(
  nrows = 15,
  ncols = 20,
  lines = 270,
  checks = 4,
  plotNumber = 101,
  kindExpt = "SUDC",
  planter = "serpentine",
  seed = 1987,
  exptName = "20WRY1",
  locationNames = "MINOT"
)
spatd$infoDesign
spatd$layoutRandom
spatd$plotsNumber
head(spatd$fieldBook, 12)

# Example 2: Generates a spatial decision block diagonal arrangement design in one location
# with 720 treatments allocated in 5 experiments or blocks for a field with dimensions
# 30 rows x 26 cols in a serpentine arrangement. In this case, we show how to set up the data
# option with the entries list.
checks <- 5; expts <- 5
list_checks <- paste("CH", 1:checks, sep = "")
treatments <- paste("G", 6:725, sep = "")
treatment_list <- data.frame(list(ENTRY = 1:725, NAME = c(list_checks, treatments)))
head(treatment_list, 12)
tail(treatment_list, 12)
spatDB <- diagonal_arrangement(
  nrows = 30,
  ncols = 26,
  checks = 5,
  plotNumber = 1,
  kindExpt = "DBUDC",
  planter = "serpentine",
  splitBy = "row",
  blocks = c(150,155,95,200,120),
  data = treatment_list
)
spatDB$infoDesign
spatDB$layoutRandom
spatDB$plotsNumber
do_optim

Generate the sparse or p-rep allocation to multiple locations.

Description

Generate the sparse or p-rep allocation to multiple locations.

Usage

do_optim(
  design = "sparse", 
  lines, 
  l, 
  copies_per_entry, 
  add_checks = FALSE, 
  checks = NULL, 
  rep_checks = NULL, 
  force_balance = TRUE, 
  seed, 
  data = NULL
)

Arguments

design
  Type of experimental design. It can be prep or sparse
lines
  Number of genotypes, experimental lines or treatments.
Number of locations or sites. By default $l = 1$.

Number of copies per plant. When design is sparse then copies_per_entry should be less than $l$

Option to add checks. Optional if design = "prep"

Number of genotypes checks.

Replication for each check.

Get balanced unbalanced locations. By default force_balance = TRUE.

(optional) Real number that specifies the starting seed to obtain reproducible designs.

(optional) Data frame with 2 columns: ENTRY | NAME. ENTRY must be numeric.

A list with three elements.

- list_locs is a list with each location list of entries.
- allocation is a matrix with the allocation of treatments.
- size_locations is a data frame with one column for each location and one row with the size of the location.

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb]


sparse_example <- do_optim(
  design = "sparse",
  lines = 120,
  l = 4,
  copies_per_entry = 3,
  add_checks = TRUE,
  checks = 4,
  seed = 15
)
full_factorial

Generates a Full Factorial Design

Description

It randomly generates a full factorial design across locations.

Usage

```r
full_factorial(
  setfactors = NULL,
  reps = NULL,
  l = 1,
  type = 2,
  plotNumber = 101,
  continuous = FALSE,
  planter = "serpentine",
  seed = NULL,
  locationNames = NULL,
  factorLabels = TRUE,
  data = NULL
)
```

Arguments

- `setfactors`: Numeric vector with levels of each factor.
- `reps`: Number of replicates (full blocks).
- `l`: Number of locations. By default `l = 1`.
- `type`: Option for CRD or RCBD designs. Values are `type = 1` (CRD) or `type = 2` (RCBD). By default `type = 2`.
- `plotNumber`: Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- `continuous`: Logical for plot number continuous or not. By default `continuous = FALSE`.
- `planter`: Option for serpentine or cartesian plot arrangement. By default `planter = 'serpentine'`.
- `seed`: (optional) Real number that specifies the starting seed to obtain reproducible designs.
- `locationNames`: (optional) Names for each location.
- `factorLabels`: (optional) If `TRUE` retain the levels labels from the original data set otherwise, numeric labels will be assigned. Default is `factorLabels = TRUE`.
- `data`: (optional) Data frame with the labels of factors.
full_factorial

Value

A list with two elements.

- `infoDesign` is a list with information on the design parameters.
- `fieldBook` is a data frame with the full factorial field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References


Examples

# Example 1: Generates a full factorial with 3 factors each with 2 levels.
# This in an RCBD arrangement with 3 reps.
fullFact1 <- full_factorial(setfactors = c(2,2,2), reps = 3, l = 1, type = 2,
plotNumber = 101,
continuous = TRUE,
planter = "serpentine",
seed = 325,
locationNames = "FARGO")

fullFact1$infoDesign
head(fullFact1$fieldBook,10)

# Example 2: Generates a full factorial with 3 factors and each with levels: 2,3,
# and 2, respectively. In this case, we show how to use the option data
FACTORS <- rep(c("A", "B", "C"), c(2,3,2))
LEVELS <- c("a0", "a1", "b0", "b1", "b2", "c0", "c1")
data_factorial <- data.frame(list(FACTOR = FACTORS, LEVEL = LEVELS))
print(data_factorial)
# This in an RCBD arrangement with 5 reps in 3 locations.
fullFact2 <- full_factorial(setfactors = NULL, reps = 5, l = 3, type = 2,
plotNumber = c(101,1001,2001),
continuous = FALSE,
planter = "serpentine",
seed = 326,
locationNames = c("Loc1","Loc2","Loc3"),
data = data_factorial)

fullFact2$infoDesign
head(fullFact2$fieldBook,10)
incomplete_blocks

Generates a Resolvable Incomplete Block Design

Description
Randomly generates a resolvable incomplete block design (IBD) of characteristics (t, k, r). The randomization can be done across locations.

Usage
incomplete_blocks(
t = NULL,
k = NULL,
r = NULL,
l = 1,
plotNumber = 101,
locationNames = NULL,
seed = NULL,
data = NULL
)

Arguments
- **t**: Number of treatments.
- **k**: Size of incomplete blocks (number of units per incomplete block).
- **r**: Number of full blocks (or resolvable replicates) (also number of replicates per treatment).
- **l**: Number of locations. By default \( l = 1 \).
- **plotNumber**: Numeric vector with the starting plot number for each location. By default plotNumber = 101.
- **locationNames**: (optional) Names for each location.
- **seed**: (optional) Real number that specifies the starting seed to obtain reproducible designs.
- **data**: (optional) Data frame with label list of treatments.

Value
A list with two elements.

- **infoDesign**: is a list with information on the design parameters.
- **fieldBook**: is a data frame with the incomplete block design field book.

Author(s)
Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]
## Reference


## Examples

```r
# Example 1: Generates a resolvable IBD of characteristics (t,k,r) = (12,4,2).
# 1-resolvable IBDs
ibd1 <- incomplete_blocks(t = 12,
                         k = 4,
                         r = 2,
                         seed = 1984)
ibd1$infoDesign
head(ibd1$fieldBook)

# Example 2: Generates a balanced resolvable IBD of characteristics (t,k,r) = (15,3,7).
# In this case, we show how to use the option data.
treatments <- paste("TX-", 1:15, sep = "")
ENTRY <- 1:15
treatment_list <- data.frame(list(ENTRY = ENTRY, TREATMENT = treatments))
head(treatment_list)
ibd2 <- incomplete_blocks(t = 15,
                         k = 3,
                         r = 7,
                         seed = 1985,
                         data = treatment_list)
ibd2$infoDesign
head(ibd2$fieldBook)
```

---

### latin_square

*Generates a Latin Square Design*

### Description

Randomly generates a latin square design of up 10 treatments.

### Usage

```r
latin_square(
  t = NULL,
  reps = 1,
  plotNumber = 101,
  planter = "serpentine",
  seed = NULL,
  locationNames = NULL,
  data = NULL
)
```
Arguments

- **t**: Number of treatments.
- **reps**: Number of full resolvable squares. By default `reps = 1`.
- **plotNumber**: Starting plot number. By default `plotNumber = 101`.
- **planter**: Option for serpentine or cartesian arrangement. By default `planter = 'serpentine'`.
- **seed**: (optional) Real number that specifies the starting seed to obtain reproducible designs.
- **locationNames**: (optional) Name for the location.
- **data**: (optional) Data frame with label list of treatments.

Value

A list with information on the design parameters.

Data frame with the latin square field book.

A list with two elements:

- `infoDesign` is a list with information on the design parameters.
- `fieldBook` is a data frame with the latin square field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Thiago de Paula Oliveira[ctb] Richard Horsley [ctb]

References


Examples

```r
# Example 1: Generates a latin square design with 4 treatments and 2 reps.
latinSq1 <- latin_square(t = 4,
                           reps = 2,
                           plotNumber = 101,
                           planter = "cartesian",
                           seed = 1980)

print(latinSq1)
summary(latinSq1)
head(latinSq1$fieldBook)

# Example 2: Generates a latin square design with 5 treatments and 3 reps.
latin_data <- data.frame(list(ROW = paste("Period", 1:5, sep = ""),
                            COLUMN = paste("Cow", 1:5, sep = ""),
                            TREATMENT = paste("Diet", 1:5, sep = "")))

print(latin_data)
latinSq2 <- latin_square(t = NULL,
                         reps = 3,
                         plotNumber = 101,
                         planter = "cartesian",
                         seed = 1980)

print(latinSq2)
summary(latinSq2)
head(latinSq2$fieldBook)
```
multi_location_prep

```r
plotNumber = 101,
planter = "cartesian",
seed = 1981,
data = latin_data)

latinSq2$squares
latinSq2$plotSquares
head(latinSq2$fieldBook)
```

---

**multi_location_prep**  
*Optimized multi-location partially replicated design*

---

**Description**

Optimized multi-location partially replicated design

**Usage**

```r
multi_location_prep(
  lines,
nrows,
ncols,
l,
planter = "serpentine",
plotNumber,
desired_avg,
copies_per_entry,
checks = NULL,
rep_checks = NULL,
exptName,
locationNames,
optim_list,
seed,
data = NULL
)
```

**Arguments**

- **lines**: Number of genotypes, experimental lines or treatments.
- **nrows**: Numeric vector with the number of rows field at each location.
- **ncols**: Numeric vector with the number of columns field at each location.
- **l**: Number of locations. By default `l = 1`.
- **planter**: Option for serpentine or cartesian movement. By default planter = 'serpentine'.
- **plotNumber**: Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- **desired_avg**: (optional) Desired average of treatments across locations.
multi_location_prep

copies_per_entry
Number of total copies per treatment.

checks
Number of checks.

rep_checks
Number of replications per check.

exptName
(optional) Name of the experiment.

locationNames
(optional) Name for each location.

optim_list
(optional) A list object of class "MultiPrep" generated by do_optim() function.

seed
(optional) Real number that specifies the starting seed to obtain reproducible designs.

data
(optional) Data frame with 2 columns: ENTRY | NAME. ENTRY must be numeric.

Value
A list of class FieldHub with several elements.

- infoDesign is a list with information on the design parameters.
- layoutRandom is a matrix with the randomization layout.
- plotNumber is a matrix with the layout plot number.
- binaryField is a matrix with the binary field.
- dataEntry is a data frame with the data input.
- genEntries is a list with the entries for replicated and non-replicated parts.
- fieldBook is a data frame with field book design. This includes the index (Row, Column).
- min_pairwise_distance is a data frame with the minimum pairwise distance between each pair of locations.
- reps_info is a data frame with information on the number of replicated and non-replicated treatments at each location.
- pairsDistance is a data frame with the pairwise distances between each pair of treatments.
- treatments_with_reps is a list with the entries for the replicated part of the design.
- treatments_with_no_reps is a list with the entries for the non-replicated part of the design.
- list_locs is a list with each location list of entries.
- allocation is a matrix with the allocation of treatments.
- size_locations is a data frame with one column for each location and one row with the size of the location.

Author(s)
Didier Murillo [aut], Salvador Gezan [aut], Jean-Marc Montpetit [ctb], Ana Heilman [ctb]

References
Examples

```r
# Example 1: Generates a spatially optimized multi-location p-rep design with 142
genotypes. The number of copies per plant available for this experiment is 9.
# This experiment is carried out in 5 locations, and there are seven seeds available
# for each plant to make replications.
# In this case, we add three controls (checks) with six reps each.
# With this setup, the experiment will have 142 treatments + 3 checks = 145
# entries and the number of plots per location after the allocation process
# will be 196.
# The average genotype allocation will be 1.5 copies per location.
## Not run:
optim_multi_prep <- multi_location_prep(
  lines = 150,
  l = 5,
  copies_per_entry = 7,
  checks = 3,
  rep_checks = c(6,6,6),
  locationNames = c("LOC1", "LOC2", "LOC3", "LOC4", "LOC5"),
  seed = 1234
)
designs <- optim_multi_prep$designs
field_book_loc_1 <- designs$LOC1$fieldBook
head(field_book_loc_1, 10)
## End(Not run)
```

---

**optimized_arrangement**  
*Generates an Spatial Un-replicated Optimized Arrangement Design*

**Description**

Randomly generates a spatial un-replicated optimized arrangement design, where the distance between checks is maximized in such a way that each row and column have control plots. Note that design generation needs the dimension of the field (number of rows and columns).

**Usage**

```r
optimized_arrangement(
  nrows = NULL,
  ncols = NULL,
  lines = NULL,
  amountChecks = NULL,
  checks = NULL,
  planter = "serpentine",
  l = 1,
  plotNumber = 101,
  seed = NULL,
  exptName = NULL,
```
locationNames = NULL,
optim = TRUE,
data = NULL
)

Arguments

nrows  Number of rows in the field.
ncols  Number of columns in the field.
lines  Number of genotypes, experimental lines or treatments.
amountChecks  Integer with the amount total of checks or a numeric vector with the replicates of each check label.
checks  Number of genotypes as checks.
planter  Option for serpentine or cartesian arrangement. By default planter = 'serpentine'.
l  Number of locations. By default l = 1.
plotNumber  Numeric vector with the starting plot number for each location. By default plotNumber = 101.
seed  (optional) Real number that specifies the starting seed to obtain reproducible designs.
exptName  (optional) Name of the experiment.
locationNames  (optional) Name for each location.
optim  By default optim = TRUE.
data  (optional) Data frame with 3 columns: ENTRY | NAME | REPS.

Value

A list with five elements.

- infoDesign is a list with information on the design parameters.
- LayoutRandom is a matrix with the randomization layout.
- plotNumber is a matrix with the layout plot number.
- dataEntry is a data frame with the data input.
- genEntries is a list with the entries for replicated and no replicated part.
- fieldBook is a data frame with field book design. This includes the index (Row, Column).

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References

Examples

# Example 1: Generates a spatial unreplicated optimized arrangement design in one location
# with 120 genotypes + 20 check plots (4 checks) for a field with dimension 14 rows x 10 cols.
## Not run:
optim_unrep1 <- optimized_arrangement(
  nrows = 14,
  ncols = 10,
  lines = 120,
  amountChecks = 20,
  checks = 1:4,
  planter = "cartesian",
  plotNumber = 101,
  exptName = "20RW1",
  locationNames = "CASSELTON",
  seed = 14124
)
optim_unrep1$infoDesign
optim_unrep1$layoutRandom
optim_unrep1$plotNumber
head(optim_unrep1$fieldBook, 12)
## End(Not run)

# Example 2: Generates a spatial unreplicated optimized arrangement design in one location
# with 200 genotypes + 20 check plots (4 checks) for a field with dimension 10 rows x 22 cols.
# As example, we set up the data option with the entries list.
## Not run:
checks <- 4
list_checks <- paste("CH", 1:checks, sep = "")
treatments <- paste("G", 5:204, sep = "")
REPS <- c(5, 5, 5, 5, rep(1, 200))
treatment_list <- data.frame(list(ENTRY = 1:204, NAME = c(list_checks, treatments), REPS = REPS))
head(treatment_list, 12)
tail(treatment_list, 12)
optim_unrep2 <- optimized_arrangement(
  nrows = 10,
  ncols = 22,
  planter = "serpentine",
  plotNumber = 101,
  seed = 120,
  exptName = "20YWA2",
  locationNames = "MINOT",
  data = treatment_list
)
optim_unrep2$infoDesign
optim_unrep2$layoutRandom
optim_unrep2$plotNumber
head(optim_unrep2$fieldBook, 12)
## End(Not run)
partially_replicated  Generates a Spatial Partially Replicated Arrangement Design

Description

Randomly generates a spatial partially replicated (p-rep) design for single or multiple locations.

Usage

```r
partially_replicated(
  nrows = NULL,
  ncols = NULL,
  repGens = NULL,
  repUnits = NULL,
  planter = "serpentine",
  l = 1,
  plotNumber = 101,
  seed = NULL,
  exptName = NULL,
  locationNames = NULL,
  multiLocationData = FALSE,
  data = NULL
)
```

Arguments

- `nrows` Numeric vector with the number of rows field at each location.
- `ncols` Numeric vector with the number of columns field at each location.
- `repGens` Numeric vector with the amount genotypes to replicate.
- `repUnits` Numeric vector with the number of reps of each genotype.
- `planter` Option for serpentine or cartesian movement. By default `planter = 'serpentine'`.
- `l` Number of locations. By default `l = 1`.
- `plotNumber` Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- `seed` (optional) Real number that specifies the starting seed to obtain reproducible designs.
- `exptName` (optional) Name of the experiment.
- `locationNames` (optional) Name for each location.
- `multiLocationData` (optional) Option to pass an entry list for multiple locations. By default `multiLocationData = FALSE`.
- `data` (optional) Data frame with 3 columns: ENTRY | NAME | REPS. If `multiLocationData = TRUE` then the data must have 4 columns: LOCATION | ENTRY | NAME | REPS.
Details

This function generates and optimizes a partially replicated (p-rep) experimental design for a given set of treatments and replication levels. The design is represented by a matrix and optimized using a pairwise distance metric. The function outputs various information about the optimized design including the field layout, replicated and unreplicated treatments, and pairwise distances between treatments. Note that the design generation needs the dimension of the field (number of rows and columns).

Value

A list with several elements.

- infoDesign is a list with information on the design parameters.
- layoutRandom is a matrix with the randomization layout.
- plotNumber is a matrix with the layout plot number.
- binaryField is a matrix with the binary field.
- dataEntry is a data frame with the data input.
- genEntries is a list with the entries for replicated and non-replicated parts.
- fieldBook is a data frame with field book design. This includes the index (Row, Column).
- min_pairwise_distance is a data frame with the minimum pairwise distance between each pair of locations.
- reps_info is a data frame with information on the number of replicated and non-replicated treatments at each location.
- pairsDistance is a data frame with the pairwise distances between each pair of treatments.
- treatments_with_reps is a list with the entries for the replicated part of the design.
- treatments_with_no_reps is a list with the entries for the non-replicated part of the design.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Jean-Marc Montpetit [ctb], Richard Horsley [ctb]

References


Examples

# Example 1: Generates a spatial optimized partially replicated arrangement design in one location with 335 genotypes for a field with dimensions 15 rows x 28 cols.
# Note that there are 250 genotypes unreplicated (only one time), 85 genotypes replicated two times, and three checks 8 times each.
## Not run:
prep_deseign1 <- partially_replicated(
plot.FielDHub

Description

Draw a field layout plot for a FielDHub object.
Usage

```r
## S3 method for class 'FieldHub'
plot(x, ...)
```

Arguments

- `x`: a object inheriting from class `FieldHub`
- `...`: further arguments passed to utility function `plot_layout()`.
  - `layout`: a integer. Options available depend on the type of design and its characteristics
  - `l`: a integer to specify the location to plot.
  - `planter`: it can be `serpentine` or `cartesian`.
  - `stacked`: it can be vertical or horizontal stacked layout.

Value

- a plot object inheriting from class `fieldLayout`
- `field_book`: a data frame with the fieldbook that includes the coordinates `ROW` and `COLUMN`.

Author(s)

Didier Murillo [aut]

Examples

```r
## Not run:
# Example 1: Plot a RCBD design with 24 treatments and 3 reps.
s <- RCBD(t = 24, reps = 3, plotNumber = 101, seed = 12)
plot(s)
## End(Not run)
```

---

**print.FieldHub**

*Print a FieldHub object*

**Description**

Prints information about any FieldHub function.

**Usage**

```r
## S3 method for class 'FieldHub'
print(x, n, ...)
```
Arguments

- **x**: an object inheriting from class
- **n**: a single integer. If positive or zero, size for the resulting object: number of elements for a vector (including lists), rows for a matrix or data frame or lines for a function. If negative, all but the n last/first number of elements of x.
- **...**: further arguments passed to `head`.

Value

an object inheriting from class `FieldHub`

Author(s)

Thiago de Paula Oliveira, <thiago.paula.oliveira@alumni.usp.br> [aut], Didier Murillo [aut]

Examples

```r
# Example 1: Generates a CRD design with 5 treatments and 5 reps each.
crd1 <- CRD(t = 5, reps = 5, plotNumber = 101, seed = 1985, locationName = "Fargo")
crd1$infoDesign
print(crd1)
```

Description

Prints a plot object of class `fieldLayout`.

Usage

```r
## S3 method for class 'fieldLayout'
print(x, ...)
```

Arguments

- **x**: a plot object inheriting from class `fieldLayout`.
- **...**: unused, for extensibility.

Value

a plot object inheriting from class `fieldLayout`.

Author(s)

Didier Murillo [aut]
print.summary.FielDHub

*Print the summary of a FieldHub object*

**Description**

Print summary information on the design parameters, and data frame structure.

**Usage**

```r
## S3 method for class 'summary.FielDHub'
print(x, ...)
```

**Arguments**

- `x` an object inheriting from class FieldHub
- `...` Unused, for extensibility

**Value**

an object inheriting from class FieldHub

**Author(s)**

Thiago de Paula Oliveira, <thiago.paula.oliveira@alumni.usp.br> [aut], Didier Murillo [aut]

---

**RCBD**

*Generates a Randomized Complete Block Design (RCBD)*

**Description**

It randomly generates a randomized complete block design (RCBD) across locations.

**Usage**

```r
RCBD(
  t = NULL,
  reps = NULL,
  l = 1,
  plotNumber = 101,
  continuous = FALSE,
  planter = "serpentine",
  seed = NULL,
  locationNames = NULL,
  data = NULL
)
```
Arguments

- **t**: An integer number with total number of treatments or a vector of dimension $t$ with labels.
- **reps**: Number of replicates (full blocks) of each treatment.
- **l**: Number of locations. By default $l = 1$.
- **plotNumber**: Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- **continuous**: Logical value for plot number continuous or not. By default `continuous = TRUE`.
- **planter**: Option for serpentine or cartesian arrangement. By default `planter = 'serpentine'`.
- **seed**: (optional) Real number that specifies the starting seed to obtain reproducible designs.
- **locationNames**: (optional) Names for each location.
- **data**: (optional) Data frame with the labels of treatments.

Value

A list with five elements.

- **infoDesign**: is a list with information on the design parameters.
- **layoutRandom**: is the RCBD layout randomization for each location.
- **plotNumber**: is the plot number layout for each location.
- **fieldBook**: is a data frame with the RCBD field book design.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References


Examples

```r
# Example 1: Generates a RCBD design with 3 blocks and 20 treatments across 3 locations.
rcbd1 <- RCBD(t = LETTERS[1:20], reps = 5, l = 3,
              plotNumber = c(101,1001, 2001),
              continuous = TRUE,
              planter = "serpentine",
              seed = 1020,
              locationNames = c("FARGO", "MINOT", "CASSELTON"))
rcbd1$infoDesign
rcbd1$layoutRandom
rcbd1$plotNumber
head(rcbd1$fieldBook)
```
# Example 2: Generates a RCBD design with 6 blocks and 18 treatments in one location.
# In this case, we show how to use the option data.
treatments <- paste("ND-", 1:18, sep = "")
treatment_list <- data.frame(list(TREATMENT = treatments))
head(treatment_list)
rcbd2 <- RCBD(reps = 6, l = 1,
plotNumber = 101,
continuous = FALSE,
planter = "serpentine",
seed = 13,
locationNames = "IBAGUE",
data = treatment_list)
rcbd2$infoDesign
rcbd2$layoutRandom
rcbd2$plotNumber
head(rcbd2$fieldBook)

---

**RCBD_augmented**

*Generates an Augmented Randomized Complete Block Design (AR-CBD)*

**Description**

It randomly generates an augmented randomized complete block design across locations (ARCBD).

**Usage**

```r
RCBD_augmented(
  lines = NULL,
  checks = NULL,
  b = NULL,
  l = 1,
  planter = "serpentine",
  plotNumber = 101,
  exptName = NULL,
  seed = NULL,
  locationNames = NULL,
  repsExpt = 1,
  random = TRUE,
  data = NULL,
  nrows = NULL,
  ncols = NULL
)
```
Arguments

- `lines` Treatments, number of lines for test.
- `checks` Number of checks per augmented block.
- `b` Number of augmented blocks.
- `l` Number of locations. By default `l = 1`.
- `planter` Option for serpentine or cartesian arrangement. By default planter = 'serpentine'.
- `plotNumber` Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- `exptName` (optional) Name of experiment.
- `seed` (optional) Real number that specifies the starting seed to obtain reproducible designs.
- `locationNames` (optional) Name for each location.
- `repsExpt` (optional) Number of reps of experiment. By default `repsExpt = 1`.
- `random` Logical value to randomize treatments or not. By default `random = TRUE`.
- `data` (optional) Data frame with the labels of treatments.
- `nrows` (optional) Number of rows in the field.
- `ncols` (optional) Number of columns in the field.

Value

A list with five elements.

- `infoDesign` is a list with information on the design parameters.
- `layoutRandom` is the ARCBD layout randomization for the first location.
- `plotNumber` is the plot number layout for the first location.
- `exptNames` is the experiment names layout.
- `data_entry` is a data frame with the data input.
- `fieldBook` is a data frame with the ARCBD field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References

Examples

# Example 1: Generates an ARCBD with 6 blocks, 3 checks for each, and 50 treatments
# in two locations.
ARCBD1 <- RCBD_augmented(lines = 50, checks = 3, b = 6, l = 2,
                         planter = "cartesian",
                         plotNumber = c(1,1001),
                         seed = 23,
                         locationNames = c("FARGO", "MINOT"))

ARCBD1$infoDesign
ARCBD1$layoutRandom
ARCBD1$exptNames
ARCBD1$plotNumber
head(ARCBD1$fieldBook, 12)

# Example 2: Generates an ARCBD with 17 blocks, 4 checks for each, and 350 treatments
# in 3 locations.
# In this case, we show how to use the option data.
checks <- 4;
list_checks <- paste("CH", 1:checks, sep = "")
treatments <- paste("G", 5:354, sep = "")
treatment_list <- data.frame(list(ENTRY = 1:354, NAME = c(list_checks, treatments)))
head(treatment_list, 12)
ARCBD2 <- RCBD_augmented(lines = 350, checks = 4, b = 17, l = 3,
                          planter = "serpentine",
                          plotNumber = c(101,1001,2001),
                          seed = 24,
                          locationNames = LETTERS[1:3],
                          data = treatment_list)

ARCBD2$infoDesign
ARCBD2$layoutRandom
ARCBD2$exptNames
ARCBD2$plotNumber
head(ARCBD2$fieldBook, 12)

rectangular_lattice  Generates a Rectangular Lattice Design.

Description

It randomly generates a rectangular lattice design across locations.

Usage

rectangular_lattice(
  t = NULL,
  k = NULL,
  r = NULL,
  l = 1,
rectangular\_lattice

plotNumber = 101,
locationNames = NULL,
seed = NULL,
data = NULL
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>Number of treatments.</td>
</tr>
<tr>
<td>k</td>
<td>Size of incomplete blocks (number of units per incomplete block).</td>
</tr>
<tr>
<td>r</td>
<td>Number of blocks (full resolvable replicates).</td>
</tr>
<tr>
<td>l</td>
<td>Number of locations. By default $l = 1$.</td>
</tr>
<tr>
<td>plotNumber</td>
<td>Numeric vector with the starting plot number for each location. By default plotNumber = 101.</td>
</tr>
<tr>
<td>locationNames</td>
<td>(optional) Names for each location.</td>
</tr>
<tr>
<td>seed</td>
<td>(optional) Real number that specifies the starting seed to obtain reproducible designs.</td>
</tr>
<tr>
<td>data</td>
<td>(optional) Data frame with label list of treatments.</td>
</tr>
</tbody>
</table>

Value

A list with two elements.

- infoDesign is a list with information on the design parameters.
- fieldBook is a data frame with the rectangular lattice design field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References


Examples

# Example 1: Generates a rectangular lattice design with 6 full blocks, 4 units per IBlock (k) # and 20 treatments in one location.
rectangularLattice1 <- rectangular_lattice(t = 20, k = 4, r = 6, l = 1,
plotNumber = 101,
locationNames = "FARGO",
seed = 126)
rectangularLattice1$infoDesign
head(rectangularLattice1$fieldBook,12)

# Example 2: Generates a rectangular lattice design with 5 full blocks, 7 units per IBlock (k)
# and 56 treatments across 2 locations.
# In this case, we show how to use the option data.
treatments <- paste("ND-", 1:56, sep = "")
ENTRY <- 1:56
treatment_list <- data.frame(list(ENTRY = ENTRY, TREATMENT = treatments))
head(treatment_list)
rectangularLattice2 <- rectangular_lattice(t = 56, k = 7, r = 5, l = 2,
plotNumber = c(1001,2001),
locationNames = c("Loc1", "Loc2"),
seed = 127,
data = treatment_list)
rectangularLattice2$infoDesign
head(rectangularLattice2$fieldBook,12)

---

## row_column

Generates a Resolvable Row-Column Design (RowColD)

### Description

It randomly generates a resolvable row-column designs (RowColD). Note that design optimization is only done at the level of rows and not columns; hence, design is suboptimal. The randomization can be done across locations.

### Usage

```r
row_column(
  t = NULL,
  nrows = NULL,
  r = NULL,
  l = 1,
  plotNumber = 101,
  locationNames = NULL,
  seed = NULL,
  data = NULL
)
```

### Arguments

- **t**  
  Number of treatments.

- **nrows**  
  Number of rows of a full resolvable replicate.

- **r**  
  Number of blocks (full resolvable replicates).

- **l**  
  Number of locations. By default l = 1.

- **plotNumber**  
  Numeric vector with the starting plot number for each location. By default plotNumber = 101.

- **locationNames**  
  (optional) Names for each location.
seed (optional) Real number that specifies the starting seed to obtain reproducible designs.
data (optional) Data frame with label list of treatments

Value
A list with four elements.

- `infoDesign` is a list with information on the design parameters.
- `resolvableBlocks` a list with the resolvable row columns blocks.
- `concurrence` is the concurrence matrix.
- `fieldBook` is a data frame with the row-column field book.

Author(s)
Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References

Examples

```r
# Example 1: Generates a row-column design with 3 full blocks and 36 treatments
# and 6 rows. This for one location.
rowcold1 <- row_column(t = 36, nrows = 6, r = 3, l = 1,
plotNumber= 101,
locationNames = "Loc1",
seed = 21)
rowcold1$infoDesign
rowcold1$resolvableBlocks
head(rowcold1$fieldBook,12)

# Example 2: Generates a row-column design with 3 full blocks and 30 treatments
# and 5 rows, for one location.
# In this case, we show how to use the option data.
treatments <- paste("ND-", 1:30, sep = "")
ENTRY <- 1:30
treatment_list <- data.frame(list(ENTRY = ENTRY, TREATMENT = treatments))
head(treatment_list)
rowcold2 <- row_column(t = 30, nrows = 5, r = 3, l = 1,
plotNumber= c(101,1001),
locationNames = c("A", "B"),
seed = 15,
data = treatment_list)
rowcold2$infoDesign
rowcold2$resolvableBlocks
head(rowcold2$fieldBook,12)
```
**run_app**

*Run the Shiny Application*

**Description**

Run the Shiny Application

**Usage**

```r
run_app(...)```

**Arguments**

... Unused, for extensibility

**Value**

A shiny app object

---

**sparse_allocation**

*Unreplicated designs using the sparse allocation approach*

**Description**

Unreplicated designs using the sparse allocation approach

**Usage**

```r
sparse_allocation(
  lines,
  nrows,
  ncols,
  l,
  planter = "serpentine",
  plotNumber,
  copies_per_entry,
  checks = NULL,
  exptName = NULL,
  locationNames,
  sparse_list,
  seed,
  data = NULL
)
```
sparse_allocation

Arguments

lines Number of genotypes, experimental lines or treatments.
nrows Number of rows in the field.
ncols Number of columns in the field.
l Number of locations or sites. By default l = 1.
planter Option for serpentine or cartesian plot arrangement. By default planter = 'serpentine'.
plotNumber Numeric vector with the starting plot number for each location. By default plotNumber = 101.
copies_per_entry Number of copies per plant. When design is sparse then copies_per_entry < 1
checks Number of genotypes checks.
exptName (optional) Name of the experiment.
locationNames (optional) Names each location.
sparse_list (optional) A class "Sparse" object generated by do_optim() function.
seed (optional) Real number that specifies the starting seed to obtain reproducible designs.
data (optional) Data frame with 2 columns: ENTRY | NAME . ENTRY must be numeric.

Value

A list with four elements.

• designs is a list with each location unreplicated randomization.
• list_locs is a list with each location list of entries.
• allocation is a matrix with the allocation of treatments.
• size_locations is a data frame with one column for each location and one row with the size of the location.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb]

References

Examples

```r
sparse <- sparse_allocation(
  lines = 120,
  l = 4,
  copies_per_entry = 3,
  checks = 4,
  locationNames = c("LOC1", "LOC2", "LOC3", "LOC4", "LOC5"),
  seed = 1234
)
```

# split_families

**Split a population of genotypes randomly into several locations.**

## Description

Split a population of genotypes randomly into several locations, with the aim of having approximately the same number of replicates of each genotype, line or treatment per location.

## Usage

```r
split_families(l = NULL, data = NULL)
```

## Arguments

- `l`: Number of locations.
- `data`: Data frame with the entry (ENTRY) and the labels of each treatment (NAME) and number of individuals per family group (FAMILY).

## Value

A list with two elements.

- `rows`: Each list is a table with a summary of cases.
- `data_locations`: is a data frame with the entries for each location.

## Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

## Examples

```
# Example 1: Split a population of 3000 and 200 families into 8 locations.
# Original dataset is been simulated.
set.seed(77)
N <- 2000; families <- 100
ENTRY <- 1:N
NAME <- paste0("SB-", 1:N)
```
FAMILY <- vector(mode = "numeric", length = N)
x <- 1:N
for (i in x) { FAMILY[i] <- sample(1:families, size = 1, replace = TRUE) }
gen.list <- data.frame(list(ENTRY = ENTRY, NAME = NAME, FAMILY = FAMILY))
head(gen.list)

# Now we are going to use the split_families() function.
split_population <- split_families(l = 8, data = gen.list)
print(split_population)
summary(split_population)
head(split_population$data_locations,12)

---

split_plot

Generates a Split Plot Design

Description

It randomly generates a split plot design (SPD) across locations.

Usage

split_plot(
  wp = NULL,
  sp = NULL,
  reps = NULL,
  type = 2,
  l = 1,
  plotNumber = 101,
  seed = NULL,
  locationNames = NULL,
  factorLabels = TRUE,
  data = NULL
)

Arguments

wp Number of whole plots, as an integer or a vector.
sp Number of sub plots per whole plot, as an integer or a vector.
reps Number of blocks (full replicates).
type Option for CRD or RCBD designs. Values are type = 1 (CRD) or type = 2 (RCBD). By default type = 2.
l Number of locations. By default l = 1.
plotNumber Numeric vector with the starting plot number for each location. By default plotNumber = 101.
seed (optional) Real number that specifies the starting seed to obtain reproducible designs.
locationNames  (optional) Names for each location.
factorLabels   (optional) If TRUE retain the levels labels from the original data set otherwise, numeric labels will be assigned. Default is factorLabels = TRUE.
data          (optional) Data frame with label list of treatments.

Value
A list with two elements.

- infoDesign is a list with information on the design parameters.
- fieldBook is a data frame with the split plot field book.

Author(s)
Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References

Examples

# Example 1: Generates a split plot design SPD with 4 whole plots, 2 sub plots per whole plot, # and 4 reps in an RCBD arrangement. This in for a single location.
SPDExample1 <- split_plot(wp = 4, sp = 2, reps = 5, l = 1, 
plotNumber = 101, 
seed = 14, 
type = 2, 
locationNames = "FARGO")

SPDExample1$infoDesign
SPDExample1$layoutlocations
head(SPDExample1$fieldBook,12)

# Example 2: Generates a split plot design SPD with 5 whole plots # (4 types of fungicide + one control), 10 sub plots per whole plot (10 bean varieties), # and 6 reps in an RCBD arrangement. This in 3 locations or sites. # In this case, we show how to use the option data.
wp <- c("NFung", paste("Fung", 1:4, sep = "")) # Fungicides (5 Whole plots)
sp <- paste("Beans", 1:10, sep = "") # Beans varieties (10 sub plots)
split_plot_Data <- data.frame(list(WHOLPLOT = c(wp, rep(NA, 5)), SUBPLOT = sp))
head(split_plot_Data, 12)
SPDExample2 <- split_plot(reps = 6, l = 3, 
plotNumber = c(101, 1001, 2001), 
seed = 23, 
type = 2, 
locationNames = c("A", "B", "C"), 
data = split_plot_Data)

SPDExample2$infoDesign
SPDExample2$layoutlocations
split_split_plot

head(SPDExample2$fieldBook, 12)

---

**split_split_plot** *Generates a Split Split Plot Design*

**Description**

It randomly generates a split plot design (SSPD) across locations.

**Usage**

```r
split_split_plot(
  wp = NULL,
  sp = NULL,
  ssp = NULL,
  reps = NULL,
  type = 2,
  l = 1,
  plotNumber = 101,
  seed = NULL,
  locationNames = NULL,
  factorLabels = TRUE,
  data = NULL
)
```

**Arguments**

- `wp` Number of whole plots, as an integer or a vector.
- `sp` Number of sub plots per whole plot, as an integer or a vector.
- `ssp` Number of sub-sub plots, as an integer or a vector.
- `reps` Number of blocks (full replicates).
- `type` Option for CRD or RCBD designs. Values are `type = 1` (CRD) or `type = 2` (RCBD). By default `type = 2`.
- `l` Number of locations. By default `l = 1`.
- `plotNumber` Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- `seed` (optional) Real number that specifies the starting seed to obtain reproducible designs.
- `locationNames` (optional) Names for each location.
- `factorLabels` (optional) If TRUE retain the levels labels from the original data set otherwise, numeric labels will be assigned. Default is `factorLabels = TRUE`.
- `data` (optional) Data frame with label list of treatments.
Value

A list with two elements.

- `infoDesign` is a list with information on the design parameters.
- `fieldBook` is a data frame with the split split plot field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References


Examples

# Example 1: Generates a split split plot design SSPD with 5 whole plots, 2 sub-plots, # 3 sub-sub plots, and 3 reps in an RCBD arrangement. This is for one location.
SSPD1 <- split_split_plot(wp = 4, sp = 2, ssp = 3, reps = 5, l = 1,
plotNumber = 101,
seed = 23,
type = 2,
locationNames = "FARGO")
SSPD1$infoDesign
head(SSPD1$fieldBook,12)

# Example 2: Generates a split split plot design SSPD with 2 whole plots, 5 sub-plots, and 10 sub-sub plots, (4 types of fungicide + one control), and 4 reps in an RCBD arrangement.
# This is for 3 locations. In this case, we show how to use the option data.
wper = paste("IRR", c("NO", "Yes"), sep = "") # Irrigation (2 Whole plots)
spp <- paste("Fung", 1:4, sep = "") # Fungicides (5 Sub plots)
ssp <- paste("Beans", 1:10, sep = "") # Beans varieties (10 Sub-sub plots)

split_split_plot_Data <- data.frame(list(WHOLPLOT = c(wp, rep(NA, 8)),
SUBPLOT = c(sp, rep(NA, 5)),
SUB_SUBPLOTS = ssp))

head(split_split_plot_Data, 10)
SSPD2 <- split_split_plot(reps = 4, l = 3,
plotNumber = c(101, 1001, 2001),
seed = 23,
type = 2,
locationNames = c("A", "B", "C"),
data = split_split_plot_Data)
SSPD2$infoDesign
head(SSPD2$fieldBook,12)
square_lattice  Generates a Square Lattice Design.

Description

It randomly generates a square lattice design across locations.

Usage

```r
square_lattice(
  t = NULL,
  k = NULL,
  r = NULL,
  l = 1,
  plotNumber = 101,
  locationNames = NULL,
  seed = NULL,
  data = NULL
)
```

Arguments

- `t` Number of treatments.
- `k` Size of incomplete blocks (number of units per incomplete block).
- `r` Number of blocks (full resolvable replicates).
- `l` Number of locations. By default `l = 1`.
- `plotNumber` Numeric vector with the starting plot number for each location. By default `plotNumber = 101`.
- `locationNames` (optional) Names for each location.
- `seed` (optional) Real number that specifies the starting seed to obtain reproducible designs.
- `data` (optional) Data frame with label list of treatments.

Value

A list with two elements.

- `INFOdesign` is a list with information on the design parameters.
- `fieldBook` is a data frame with the square lattice design field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]
References


Examples

# Example 1: Generates a square lattice design with 5 full blocks, 8 units per IBlock,
# 8 IBlocks for a square number of treatments of 64 in two locations.
squareLattice1 <- square_lattice(t = 64, k = 8, r = 5, l = 2,
plotNumber = c(1001, 2001),
locationNames = c("FARGO", "MINOT"),
seed = 1986)

squareLattice1$infoDesign
head(squareLattice1$fieldBook,12)

# Example 2: Generates a square lattice design with 3 full blocks, 7 units per IBlock,
# 7 IBlocks for a square number of treatments of 49 in one location.
# In this case, we show how to use the option data.
treatments <- paste("G", 1:49, sep = "")
ENTRY <- 1:49
treatment_list <- data.frame(list(ENTRY = ENTRY, TREATMENT = treatments))
head(treatment_list)
squareLattice2 <- square_lattice(t = 49, k = 7, r = 3, l = 1,
plotNumber = 1001,
locationNames = "CASSELTON",
seed = 1986,
data = treatment_list)

squareLattice2$infoDesign
head(squareLattice2$fieldBook,12)

---

strip_plot

Strip Plot Design

Description

It randomly generates a strip plot design across locations.

Usage

strip_plot(
  Hplots = NULL,
  Vplots = NULL,
  b = 1,
  l = 1,
  plotNumber = NULL,
  planter = "serpentine",
  locationNames = NULL,
  seed = NULL,
strip_plot

factorLabels = TRUE, 
data = NULL
)

Arguments

Hplots Number of horizontal factors, as an integer or a vector.
Vplots Number of vertical factors, as an integer or a vector.
b Number of blocks (full replicates).
l Number of locations. By default l = 1.
plotNumber Numeric vector with the starting plot number for each location. By default plotNumber = 101.
planter Option for serpentine or cartesian arrangement. By default planter = 'serpentine'.
locationNames (optional) Names for each location.
seed (optional) Real number that specifies the starting seed to obtain reproducible designs.
factorLabels (optional) If TRUE retain the levels labels from the original data set otherwise, numeric labels will be assigned. Default is factorLabels = TRUE.
data (optional) data frame with the labels of vertical and horizontal plots.

Value

A list with four elements.

- infoDesign is a list with information on the design parameters.
- stripsBlockLoc is a list with the strip blocks for each location.
- plotLayouts is a list with the layout plot numbers for each location.
- fieldBook is a data frame with the strip plot field book.

Author(s)

Didier Murillo [aut], Salvador Gezan [aut], Ana Heilman [ctb], Thomas Walk [ctb], Johan Aparicio [ctb], Richard Horsley [ctb]

References


Examples

# Example 1: Generates a strip plot design with 5 vertical strips and 4 horizontal strips,
# with 3 reps in one location.
H <- paste("H", 1:4, sep = "")
V <- paste("V", 1:5, sep = "")
strip1 <- strip_plot(Hplots = H,
                      Vplots = V)
summary.FielDHub

Summary a FielDHub object

Description

Summarise information on the design parameters, and data frame structure

Usage

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

Arguments

object an object inheriting from class FielDHub
...
   Unused, for extensibility
swap_pairs

Value

an object inheriting from class summary.FielDHub

Author(s)

Thiago de Paula Oliveira, <thiago.paula.oliveira@alumni.usp.br>

Examples

# Example 1: Generates a CRD design with 5 treatments and 5 reps each.
crd1 <- CRD(t = 5, reps = 5, plotNumber = 101, seed = 1985, locationName = "Fargo")
crd1$infoDesign
summary(crd1)

---

swap_pairs

Swap pairs in a matrix of integers

Description

Modifies the input matrix X to ensure that the distance between any two occurrences of the same integer is at least a dist d, by swapping one of the occurrences with a random occurrence of a different integer that is at least d away. The function starts with starting_dist = 3 and increases it by 1 until the algorithm no longer converges or stop_iter iterations have been performed.

Usage

swap_pairs(X, starting_dist = 3, stop_iter = 50)

Arguments

X 
A matrix of integers.

starting_dist 
The minimum starting distance to enforce between pairs of occurrences of the same integer. Default is 3.

stop_iter 
The maximum number of iterations to perform. Default is 100.

Value

A list containing the following elements:

optim_design 
The modified matrix.

designs 
A list of all intermediate designs, starting from the input matrix.

distances 
A list of all pair distances for each intermediate design.

min_distance 
An integer indicating the minimum distance between pairs of occurrences of the same integer.

pairwise_distance 
A data frame with the pairwise distances for the final design.
Author(s)

Jean-Marc Montpetit [aut], Didier Murillo [aut]

Examples

# Create a matrix X with the numbers 1 to 10 are twice and 11 to 50 are once.
# The matrix has 6 rows and 10 columns
set.seed(123)
X <- matrix(sample(c(rep(1:10, 2), 11:50), replace = FALSE), ncol = 10)
X

# Swap pairs
B <- swap_pairs(X, starting_dist = 3)
B$optim_design
B$designs
B$distances
Index

alpha_lattice, 2
CRD, 4
diagonal_arrangement, 6
do_optim, 9
full_factorial, 11
head, 25
incomplete_blocks, 13
latin_square, 14
multi_location_prep, 16
optimized_arrangement, 18
partially_replicated, 21
plot.FieldHub, 23
print.FieldHub, 24
print.fieldLayout, 25
print.summary.FieldHub, 26
RCBD, 26
RCBD_augmented, 28
rectangular_lattice, 30
row_column, 32
run_app, 34
sparse_allocation, 34
split_families, 36
split_plot, 37
split_split_plot, 39
square_lattice, 41
strip_plot, 42
summary.FieldHub, 44
swap_pairs, 45