Package ‘statgenSTA’

October 14, 2024

Type Package

Title Single Trial Analysis (STA) of Field Trials

Version 1.0.14

Date 2024-10-11

Description Phenotypic analysis of field trials using mixed models with and without spatial components. One of a series of statistical genetic packages for streamlining the analysis of typical plant breeding experiments developed by Biometris.
Some functions have been created to be used in conjunction with the R package 'asreml' for the 'ASReml' software, which can be obtained upon purchase from 'VSN' international (<https://vsni.co.uk/software/asreml-r/>).

License GPL

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Depends R (>= 3.3)

Imports emmeans, ggplot2 (>= 3.3.0), ggrepel, gridExtra, knitr, lme4, mapproj, maps, qtl, rlang, scales (>= 1.1.0), SpATS (>= 1.0-18), xtable

Suggests asreml(>= 3.0), rmarkdown, testthat, tibble

SystemRequirements pdflatex

VignetteBuilder knitr

URL https://biometris.github.io/statgenSTA/index.html, https://github.com/Biometris/statgenSTA/

BugReports https://github.com/Biometris/statgenSTA/issues

NeedsCompilation no

Author Bart-Jan van Rossum [aut, cre] (<https://orcid.org/0000-0002-8673-2514>), Fred van Eeuwijk [ctb] (<https://orcid.org/0000-0003-3672-2921>), Martin Boer [ctb] (<https://orcid.org/0000-0002-1879-4588>),

Marcos Malosetti [ctb] (<https://orcid.org/0000-0002-8150-1397>),
Daniela Bustos-Korts [ctb] (<https://orcid.org/0000-0003-3827-6726>),
Emilie J. Millet [ctb] (<https://orcid.org/0000-0002-2913-4892>),
Joao Paulo [ctb] (<https://orcid.org/0000-0002-4180-0763>),
Maikel Verouden [ctb] (<https://orcid.org/0000-0002-4893-3323>),
Willem Kruijer [ctb] (<https://orcid.org/0000-0001-7179-1733>),
Ron Wehrens [ctb] (<https://orcid.org/0000-0002-8798-5599>),
Choazhi Zheng [ctb] (<https://orcid.org/0000-0001-6030-3933>)

Maintainer Bart-Jan van Rossum <bart-jan.vanrossum@wur.nl>

Repository CRAN

Date/Publication 2024-10-14 07:40:02 UTC

Contents

dropsRaw 2
extractSTA 4
fitTD 6
getMeta 10
outlierSTA 12
plot.STA 14
plot.TD 16
report 20
report.STA 20
STAtoCross 22
STAtoTD 23
summary.STA 25
summary.TD 26
TD 28
TDHeat05 31
TDMaize 32

Index **34**

`dropsRaw` *DROPS data set*

Description

This dataset comes from the European Union project DROPS (DROught-tolerant yielding PlantS). A panel of 256 maize hybrids was grown with two water regimes (irrigated or rainfed), in seven fields in 2012 and 2013, respectively, spread along a climatic transect from western to eastern Europe, plus one site in Chile in 2013. This resulted in 28 experiments defined as the combination of one year, one site and one water regime, with two and three repetitions for rainfed and irrigated treatments, respectively. A detailed environmental characterisation was carried out, with hourly records of micrometeorological data and soil water status, and associated with precise measurement of phenology. Grain yield and its components were measured at the end of the experiment.

10 experiments have been selected from the full data set, two for each of the five main environmental scenarios that were identified in the data. The scenarios have been added to the data as well as a classification of the genotypes in four genetic groups.

The data.frame contains the raw phenotypic data per experiment (Location × year × water regime), one value per individual plot and outliers have been removed.

A data.frame with 6499 rows and 25 columns.

Experiment experiments ID described by the three first letters of the city's name followed by the year of experiment and the water regime with W for watered and R for rain-fed.

Site location where the experiment was performed

year year in which the experiment was performed

plotId plot identifier (when available)

treatment targeted water regime

Code_ID, Variety_ID, Accession_ID identifier of the genotype

Replicate, block experimental design factors

Row, Column 2D coordinates of each plot

grain.yield genotypic mean for yield adjusted at 15\ in ton per hectare (t ha^-1)

grain.number genotypic mean for number of grain per square meter

grain.weight genotypic mean for individual grain weight in milligram (mg)

anthesis genotypic mean for male flowering (pollen shed), in thermal time cumulated since emergence (d_20°C)

silking genotypic mean for female flowering (silking emergence), in thermal time cumulated since emergence (d_20°C)#'

plant.height genotypic mean for plant height, from ground level to the base of the flag leaf (highest) leaf in centimeter (cm)

tassel.height genotypic mean for plant height including tassel, from ground level to the highest point of the tassel in centimeter (cm)

ear.height genotypic mean for ear insertion height, from ground level to ligule of the highest ear leaf in centimeter (cm)

Lat The latitude of the location where the experiment was performed

Long The longitude of the location where the experiment was performed

scenarioWater water scenario for the experiment, well watered (WW) or water deficit (WD)

scenarioTemp temperature scenario for the experiment, Cool, Hot or Hot(Day)

scenarioFull the full scenario for the experiment, a combination of scenarioWater and scenarioTemp

geneticGroup the genetic group to which the genotype belongs

Usage

```
dropsRaw
```

Format

An object of class `data.frame` with 6486 rows and 26 columns.

Source

doi:10.15454/IASSTN

References

Millet, E. J., Pommier, C., et al. (2019). A multi-site experiment in a network of European fields for assessing the maize yield response to environmental scenarios (Data set). doi:10.15454/IASSTN

`extractSTA` *Extract statistics from fitted models*

Description

Extract and calculate various results for fitted models such as BLUEs, BLUPs, unit errors and heritabilities. For a full list of results that can be extracted, see the table below.

The result(s) to extract can be specified in `what`.
If a single result is extracted, this result is returned as a `data.frame`, If this is not possible, because the format of the result is incompatible with the `data.frame` format, the result is returned as a list. E.g. if BLUEs are extracted, the output of the function is a `data.frame` with BLUEs. However if varCompF is extracted, the output of the function is a `list`.

Results that are returned as `data.frame` are marked as such in the asDataFrame column in the table. If the default return value for a result is a `data.frame` that can be overridden by the user by specifying `asDataFrame = FALSE`. When doing so the result will be returned as a list of data.frames, one per trial. They other way round is not possible. If a result is returned as a `list` according to the table, it cannot be returned as a `data.frame`.
If multiple results are extracted at the same time, these are always returned as a list.

Most results can only be calculated if a model is fitted with genotype as fixed or with genotype as random. E.g. to compute heritabilities a model should be fitted with genotype as random effect. This is indicated in the table in the column model with "F" and "R" respectively.

Possible options for `what` are:

result	model	description	asDataFrame
BLUEs	F	Best Linear Unbiased Estimators	yes
seBLUEs	F	standard errors of the BLUEs	yes
ue	F	unit errors - only for lme4 and asreml	yes
varCompF	F	variance components for the model with genotype as fixed component	
fitted	F	fitted values for the model with genotype as fixed component	yes
residF	F	residuals for the model with genotype as fixed component	yes
stdResF	F	standardized residuals for the model with genotype as fixed component	yes

wald	F	results of the wald test - only for lme4 and asreml	
CV	F	Coefficient of Variation	yes
rDfF	F	residual degrees of freedom for the model with genotype as fixed component	yes
sed	F	standard error of difference - only for asreml	
lsd	F	least significant difference - only for asreml	
BLUPs	R	Best Linear Unbiased Predictors	yes
seBLUPs	R	standard errors of the BLUPs	yes
heritability	R	generalized heritability	yes
varCompR	R	variance components for the model with genotype as random component	
varGen	R	genetic variance component	yes
varErr	R	residual variance component	yes
varSpat	R	spatial variance components - only for SpATS	
rMeans	R	fitted values for the model with genotype as random component	yes
ranEf	R	random genetic effects	yes
residR	R	residuals for the model with genotype as random component	yes
stdResR	R	standardized residuals for the model with genotype as random component	yes
rDfR	R	residual degrees of freedom for the model with genotype as random component	yes
effDim	R	effective dimensions - only for SpATS	
ratEffDim	R	ratios of the effective dimensions - only for SpATS	

Usage

```
extractSTA(
  STA,
  trials = names(STA),
  traits = NULL,
  what = "all",
  asDataFrame = length(what) == 1 && what != "all",
  keep = NULL,
  restoreColNames = FALSE
)
```

Arguments

`STA`	An object of class STA.
`trials`	A character vector of trials for which the statistics should be computed. If not supplied, statistics are computed for all trials that have been modeled.
`traits`	A character vector of traits for which the statistics should be computed. If not supplied, statistics are computed for all traits that have been modeled.
`what`	A character vector indicating which statistics should be computed. Most statistics are available for all models, some only for models fitted using a certain engine. If this is the case, this is indicated in the list with options in details. If `what = "all"`, all available statistics are computed.
`asDataFrame`	Should the output be reshaped to a data.frame. This is only possible if the number of statistics to extract is one.
`keep`	A character vector of column(s) in the object of class `TD` used for modeling. These columns will be kept as output when computing fitted values, residuals,

standardized residuals and rMeans. Columns can also be kept when computing (se)BLUEs and (se)BLUPs but only if the column to keep contains unique values for the modeled variables, i.e. a column repId with several different values per genotype cannot be kept.

`restoreColNames`
: Should the original column names be restored in the output of the extracted data?

Value

Depending on the input either a data.frame or a list with, per trial for which statistics have been extracted, a list of those statistics.

See Also

fitTD

Examples

```
## Fit model using SpATS.
modSp <- fitTD(TD = TDHeat05,
               design = "res.rowcol",
               traits = "yield")

## Extract all available statistics from the fitted model.
extr <- extractSTA(modSp)

## Extract only the BLUEs from the fitted model.
BLUEs <- extractSTA(modSp,
                    what = "BLUEs")

## Extract only the BLUEs from the fitted model and keep trial as variable in
## the output.
BLUEs2 <- extractSTA(modSp,
                     what = "BLUEs",
                     keep = "trial")
```

`fitTD` *Fit single trial mixed model*

Description

Perform REML analysis given a specific experimental design using either SpATS, lme4 or asreml. SpATS is used as a default method when row coordinates (rowCoord) and column coordinates (colCoord) are present, lme4 otherwise. See details for the exact models fitted.

Usage

```
fitTD(
  TD,
  trials = names(TD),
  design = NULL,
  traits,
  what = c("fixed", "random"),
  covariates = NULL,
  useCheckId = FALSE,
  spatial = FALSE,
  engine = NA,
  control = NULL,
  progress = FALSE,
  ...
)
```

Arguments

TD	An object of class TD.
trials	A character vector specifying the trials for which the models should be fitted.
design	A character string specifying the experimental design. Either "ibd" (incomplete block design), "res.ibd" (resolvable incomplete block design), "rcbd" (randomized complete block design), "rowcol" (row column design) or "res.rowcol" (resolvable row column design). Can be ignored when the trial design is specified in the meta data (see setMeta).
traits	A character vector specifying the traits for which the models should be fitted.
what	A character vector specifying whether "genotype" should be fitted as fixed or random effect. If not specified, both models are fitted.
covariates	A character vector specifying covariates to be fitted as extra fixed effects in the model.
useCheckId	Should checkId be used as a fixed effect in the model? If TRUE, TD has to contain a column 'checkId'. Using checkId as fixed effect can only be done when genotype is fitted as a random effect in the model.
spatial	Should spatial models be tried? Spatial models can only be fitted with SpATS and asreml. If SpATS is used for modeling, only spatial models can be fitted and spatial is always set to TRUE. If asreml is used, fitting spatial models is optional.
engine	A character string specifying the name of the mixed modeling engine to use, either "SpATS", "lme4" or "asreml." For spatial models, "SpaTS" is used as default, for other models "lme4".
control	An optional list with control parameters to be passed to the actual fitting functions. Currently nSeg and nestDiv are valid parameters when fitting a model using SpATS. They pass a value to nseg and nest.div in PSANOVA respectively. For nSeg also a named list can be supplied containing values for nSeg per trial. criterion is a valid parameter when fitting a spatial model using asreml. It may be used to pass a goodness-of-fit criterion for comparing different spatial models. See also in details. Other parameters are ignored.

progress	Should the progress of the modeling be printed. If TRUE, for every trial a line is output indicating the traits fitted for the particular trial.
...	Further arguments to be passed to SpATS, lme4 or asreml.

Details

The actual model fitted depends on the design. For the supported designs, the following models are used:

Design	Code	Model fitted
incomplete block design	ibd	trait = **subBlock** + genotype + ϵ
resolvable incomplete block design	res.ibd	trait = *repId* + **repId:subBlock** + genotype + ϵ
randomized complete block design	rcbd	trait = *repId* + genotype + ϵ
row column design	rowcol	trait = **rowId** + **colId** + genotype + ϵ
resolvable row column design	res.rowcol	trait = *repId* + **repId:rowId** + **repId:colId** + genotype + ϵ

In the models above, fixed effects are indicated in *italics* whereas random effects are indicated in **bold**. genotype can be fitted as fixed or as random effect depending on the value of the parameter what. Extra fixed effects may be fitted using the parameter covariates.

If SpATS is used as modeling engine, an extra spatial term is always included in the model. This term is constructed using the function PSANOVA from the SpATS package as
PSANOVA(colCoord, rowCoord, nseg = nSeg, nest.div = 2) where
nSeg = (number of columns / 2, number of rows / 2). nseg and nest.div can be modified using the control parameter.

When asreml is used for modeling and spatial is TRUE seven models are fitted with different random terms and covariance structure. The best model is determined based on a goodness-of-fit criterion, either AIC or BIC. This can be set using the control parameter criterion, default is AIC. The fitted random terms depend on the structure of the data. If the trial has a regular structure, i.e. all replicates appear the same amount of times in the trial, the following combinations of random and spatial terms are fitted:

Random part	Spatial part
random effects based on design	none
random effects based on design	AR1(rowId):colId
random effects based on design	rowId:AR1(colId)
random effects based on design	AR1(rowId):ar1(colId)
random effects based on design + nugget	AR1(rowId):colId
random effects based on design + nugget	rowId:AR1(colId)
random effects based on design + nugget	AR1(rowId):AR1(colId)

If the design is not regular the following combinations of random and spatial terms are fitted:

Random part	Spatial part
random effects based on design	none

random effects based on design	exp(rowCoord):colCoord
random effects based on design	rowCoord:exp(colCoord)
random effects based on design	iexp(rowCoord, colCoord)
random effects based on design + nugget	exp(rowCoord):colCoord
random effects based on design + nugget	rowCoord:exp(colCoord)
random effects based on design + nugget	iexp(rowCoord,colCoord)

Value

An object of class `STA`, a list containing, per trial that has been analyzed, a list of:

`mRand`	A list of models with fitted with genotype as random effect.
`mFix`	A list of models fitted with genotype as fixed effect.
`TD`	An object of class `TD` containing the data on which `mRand` and `mFix` are based.
`traits`	A character vector indicating the traits for which the models are fitted.
`design`	A character string containing the design of the trial. (see `fitTD` for the possible designs).
`spatial`	A character string indicating the spatial part of the model. `FALSE` if no spatial design has been used.
`engine`	A character string containing the engine used for the analysis.
`predicted`	A character string indicating the variable that has been predicted.
`sumTab`	A data.frame with a summary table for the spatial models tried when `engine = "asreml"` and `spatial = TRUE`
`useCheckId`	A boolean indicating if checkId was used when fitting the models

References

Maria Xose Rodriguez-Alvarez, Martin P. Boer, Fred A. van Eeuwijk, Paul H.C. Eilers (2017). Correcting for spatial heterogeneity in plant breeding experiments with P-splines. Spatial Statistics doi:10.1016/j.spasta.2017.10.003

Butler, D. G., et al. (2010). Analysis of Mixed Models for S language environments: ASReml-R reference manual. Brisbane, DPI Publications

Douglas Bates, Martin Maechler, Ben Bolker, Steve Walker (2015). Fitting Linear Mixed-Effects Models Using lme4. Journal of Statistical Software, 67(1), 1-48. https://www.jstatsoft.org/article/view/v067i01/0.

Examples

```
## Fit model using lme4.
lmeMod <- fitTD(TD = TDHeat05,
                design = "ibd",
                traits = "yield",
                what = "fixed",
                engine = "lme4")

## Summarize results.
```

```
summary(lmeMod)

## Create base plots of the results.
plot(lmeMod)

## Create a pdf report summarizing results.
report(lmeMod,
       outfile = tempfile(fileext = ".pdf"),
       what = "fixed")

## Fit model using SpATS.
spaMod <- fitTD(TD = TDHeat05,
                design = "res.rowcol",
                traits = "yield",
                what = "fixed")
summary(spaMod)

## Create spatial plots of the results.
plot(spaMod, plotType = "spatial")

## Fit model using SpATS.
## Specify the number of segments to use in PSANOVA
spaMod2 <- fitTD(TD = TDHeat05,
                 design = "res.rowcol",
                 traits = "yield",
                 what = "fixed",
                 control = list(nSeg = c(13, 26)))

## Create a pdf report summarizing results.
report(spaMod, outfile = tempfile(fileext = ".pdf"), what = "fixed")

## Fit model using asreml.
if (requireNamespace("asreml", quietly = TRUE)) {
  asrMod <- fitTD(TD = TDHeat05,
                  design = "res.rowcol",
                  traits = "yield",
                  what = "fixed",
                  engine = "asreml")
  summary(asrMod)

  report(asrMod,
         outfile = tempfile(fileext = ".pdf"),
         what = "fixed")
}
```

---

getMeta *Get and set metadata for TD objects*

Description

Functions for extracting and adding metadata for objects of class TD.

`getMeta` extracts a data.frame with location, date, design, latitude, longitude, plot width and plot length for all trials in TD.

`setMeta` adds metadata from a data.frame to an object of class TD. See details for the specifications of the data.frame.

The most common use case is extracting metadata from a TD object, modifying the content and then adding it back to the TD object.
Information in the metadata of a TD object is used in plotting functions (e.g. latitude and longitude for a map plot) and when fitting models on the data (the trial design).

Usage

```
getMeta(TD)

setMeta(TD, meta)
```

Arguments

`TD`	An object of class TD.
`meta`	A data.frame containing metadata.

Details

When setting metadata, metadata has to be a data.frame with rownames corresponding to the trials in `TD`. The data.frame should contain one or more of the following columns:

trLocation The location of the trial. Used as default name when creating plots and summaries.

trDate The date of the trial.

trDesign The design of the trial. One of "none" (no (known) design), "ibd" (incomplete-block design), "res.ibd" (resolvable incomplete-block design), "rcbd" (randomized complete block design), "rowcol" (row-column design) or "res.rowcol" (resolvable row-column design). Used when fitting models.

trLat The latitude of the trial on a scale of -90 to 90. Used when plotting the trials on a map.

trLong The longitude of the trial on a scale of -180 to 180. Used when plotting the trials on a map.

trPlWidth The width of the plot. Used in combination with trPlLength to determine the size of the plots in a layout plot of a trial.

trPlLength The length of the plot. Used in combination with trPlWidth to determine the size of the plots in a layout plot of a trial.

The values of the metadata of TD will be set to the values in the corresponding column in `meta`. Existing values will be overwritten, but `NA` will be ignored so setting a value to `NA` won't result in accidentally removing it.

See Also

Other functions for TD objects: TD, plot.TD(), summary.TD()

Examples

```
data("dropsRaw")

## Create a TD object.
dropsTD <- createTD(data = dropsRaw[dropsRaw$year == 2012, ],
                    genotype = "Variety_ID",
                    trial = "Experiment",
                    loc = "Site",
                    repId = "Replicate",
                    subBlock = "block",
                    rowCoord = "Row",
                    colCoord = "Column",
                    trLat = "Lat",
                    trLong = "Long")

## Get meta data from dropsTD.
(dropsMeta <- getMeta(dropsTD))

## Add trial date to meta data.
dropsMeta$trDate <- as.Date(rep("010112", times = 5), "%d%m%y")

## Add back meta data to wheatTD.
dropsTD <- setMeta(dropsTD,
                   dropsMeta)
```

---

outlierSTA *Identifying outliers in objects of class STA*

Description

Function to identify observations with standardized residuals exceeding rLimit. If not provided rLimit is computed as qnorm(1 - 0.5 / rDf) where rDf is the residual degrees of freedom for the model. This value is then restricted to the interval 2..4. Alternatively a custom limit may be provided.
If verbose = TRUE a summary is printed of outliers and observations that have the same value for commonFactors. The column outlier in the output can be used to distinguish real outliers from observations included because of their commonFactors.

Usage

```
outlierSTA(
  STA,
  trials = NULL,
  traits = NULL,
```

```
  what = NULL,
  rLimit = NULL,
  commonFactors = NULL,
  verbose = TRUE
)
```

Arguments

STA	An object of class STA.
trials	A character vector specifying the trials for which outliers should be identified. If trials = NULL, all trials are included.
traits	A character vector specifying the traits for which outliers should be identified.
what	A character string indicating whether the outliers should be identified for the fitted model with genotype as fixed (what = "fixed") or genotype as random (what = "random") factor. If STA contains only one model this model is chosen automatically.
rLimit	A numerical value used for determining when a value is considered an outlier. All observations with standardized residuals exceeding rLimit will be marked as outliers.
commonFactors	A character vector specifying the names of columns in TD used for selecting observations that are similar to the outliers. If commonFactors = NULL, only outliers are reported and no similar observations.
verbose	Should the outliers be printed to the console?

Value

A list with two components:

- indicator - a list of numeric vectors indicating the location of the outliers in the data
- outliers - a data.frame containing the outliers and observations similar to the outliers as defined by commonFactors

Examples

```
## Fit a model using lme4.
modLme <- fitTD(TD = TDHeat05,
                traits = "yield",
                design = "res.rowcol",
                engine = "lme4")

## Detect outliers in the standardized residuals of the fitted model.
outliers <- outlierSTA(STA = modLme,
                       traits = "yield")
```

`plot.STA` *Plot function for class STA*

Description

This function draws either four base plots:

- A histogram of the residuals
- A normal Q-Q plot
- A residuals vs fitted values plot
- An absolute residuals vs fitted values plot

or five or six spatial plots:

- A spatial plot of the raw data
- A spatial plot of the fitted data
- A spatial plot of the residuals
- A spatial plot of the estimated spatial trend (SpATS only)
- A spatial plot of the BLUEs or BLUPs
- A histogram of the BLUEs or BLUPs

Spatial plots can only be made if the data contains both row and column information.

Usage

```
## S3 method for class 'STA'
plot(
  x,
  ...,
  trials = NULL,
  traits = NULL,
  what = NULL,
  plotType = c("base", "spatial"),
  spaTrend = c("raw", "percentage"),
  outCols = ifelse(plotType == "base", 2, 3),
  title = NULL,
  output = TRUE
)
```

Arguments

`x`	An object of class STA.
`...`	Further graphical parameters.
`trials`	A character vector indicating the trials to plot. If `trials = NULL`, all trials are plotted.

traits	A character vector indicating the traits to plot. If traits = NULL, all traits are plotted.
what	A character string indicating whether the fitted model with genotype as fixed (what = "fixed") or genotype as random (what = "random") factor should be plotted. If x contains only one model this model is chosen automatically.
plotType	A character string indicating whether base plots or spatial plots should be made.
spaTrend	A character string indicating how the spatial trend should be displayed. Either "raw" (original scale), or "percentage". If "percentage", the estimated spatial trend is scaled (i.e., divided by the average of the observed response variable of interest across the field) and results are shown as a percentage.
outCols	An integer indicating the number of columns to use for displaying the plots. Usually the default of 2 for base plots and 3 for spatial plots will be fine, but decreasing the numbers may help for nicer printing.
title	A character string used a title for the plot. Note that when a title is specified and multiple plots are created, all plots will get the same title.
output	Should the plot be output to the current device? If FALSE only a list of ggplot objects is invisibly returned.

Value

A list containing ggplot objects for the selected plots.

See Also

Other functions for STA objects: STAtoCross(), STAtoTD(), report.STA(), summary.STA()

Examples

```
## Run a single trait analysis using SpATS.
modSp <- fitTD(TD = TDHeat05,
               design = "res.rowcol",
               traits = "yield")

## Create base plots.
plot(modSp,
     what = "fixed",
     plotType = "base")

## Create spatial plots.
plot(modSp,
     what = "fixed",
     plotType = "spatial")

## Create spatial plots showing the spatial trend as percentage.
plot(modSp,
     what = "fixed",
     plotType = "spatial",
     spaTrend = "percentage")
```

plot.TD *Plot function for class TD*

Description

Plotting function for objects of class TD. Plots either the layout of the different trials within the TD object or locates the trials on a map. Also a boxplot can be made for selected traits and trials, a plot of correlations between trials and a scatter plot matrix. A detailed description and optional extra parameters of the different plots is given in the sections below.

Usage

```
## S3 method for class 'TD'
plot(
  x,
  ...,
  plotType = c("layout", "map", "box", "cor", "scatter"),
  trials = names(x),
  traits = NULL,
  title = NULL,
  output = TRUE
)
```

Arguments

x	An object of class TD.
...	Extra plot options. Described per plotType in their respective section.
plotType	A single character string indicating which plot should be made. See the sections below for a detailed explanation of the plots.
trials	A character vector indicating which trials to include in the plot.
traits	A character vector indicating for which traits a plot should be made. Ignored if plotType = "map".
title	A character string used a title for the plot. Note that when a title is specified and multiple plots are created, all plots will get the same title.
output	Should the plot be output to the current device? If FALSE only a list of ggplot objects is invisibly returned.

Layout Plot

Plots the layout of the selected trials. This plot can only be made for trials that contain both row (rowCoord) and column (colCoord) information. If either one of those is missing the trial is skipped with a warning. If blocks (subBlock) are available for a trial these can be colored in different colors per block by setting colorSubBlock = TRUE. If replicates (repId) are available a black line is plotted between different replicates. Missing plots are indicated in white. These can either be single plots in a trial or complete missing columns or rows.
Extra parameter options:

showGeno Should individual genotypes be indicated as text in the plot? Defaults to `FALSE`

sizeGeno The text size for indicating individual genotypes. Defaults to 2. Ignored if `showGeno = FALSE`.

highlight A character vector of genotypes to be highlighted in the plot.

colHighlight A character vector specifying colors to use for the highlighted genotypes. If not specified, default ggplot colors are used.

colorSubBlock Should blocks be colored with a different color per subBlock? Defaults to `FALSE`. `colorSubBlock` is ignored when highlight is used to highlight genotypes.

colSubBlock A character vector specifying colors to use for the subBlocks. If not specified, default ggplot colors are used.

Map Plot

A map is plotted with the locations of the trials in the TD object. Mapping the trials is done based on latitude and longitude that can be added when creating an object of class TD. Trials for which either latitude or longitude is not available are skipped with a warning message. The countries in which the trials are located will be plotted on a single map and the location of the trials will be indicated on this map. The actual plot is made using ggplot, but for getting the data for the borders of the countries the maps package is needed.
Extra parameter options:

colorTrialBy A character string indicating a column in `TD` by which the trials on the map are colored.

colTrial A character vector with plot colors for the trials. A single color when `colorTrialBy = NULL`, a vector of colors otherwise.

printTrialNames Should trial names be printed. Defaults to `TRUE`. Setting this to `FALSE` can be useful if there are many trials.

minLatRange A positive numerical value indicating the minimum range (in degrees) for the latitude on the plotted map. Defaults to 10.

minLongRange A positive numerical value indicating the minimum range (in degrees) for the longitude on the plotted map. Defaults to 5.

Box Plot

Creates a boxplot per selected trait grouped by trial. Extra parameter options:

groupBy A character string indicating a column in `TD` by which the boxes in the plot should be grouped. By default the boxes are grouped per trial.

colorTrialBy A character string indicating a column in `TD` by which the boxes are colored. Coloring will be done within the groups indicated by the `groupBy` parameter.

colTrial A character vector with plot colors for the trials. A single color when `colorTrialBy = NULL`, a vector of colors otherwise.

orderBy A character string indicating the way the boxes should be ordered. Either "alphabetic" for alphabetical ordering of the groups, "ascending" for ordering by ascending mean, or "descending" for ordering by descending mean. Default boxes are ordered alphabetically.

Correlation Plot

Draws a heat map of correlations between trials per selected trait. If genotypes are replicated within trials genotypic means are taken before computing correlations. The order of the trials in the heat map is determined by clustering them. Closely related trials will be plotted close to each other.

Scatter Plot

Draws a scatter plot matrix per selected trait. If genotypes are replicated within trials genotypic means are taken before plotting. The lower left of the matrix contains scatter plots between trials. The diagonal contains histograms of the data per trial.
Extra parameter options:

colorGenoBy A character string indicating a column in `TD` by which the genotypes in the scatter plots are colored.

colGeno A character vector with plot colors for the genotypes. A single color when `colorGenoBy = NULL`, a vector of colors otherwise.

colorTrialBy A character string indicating a column in `TD` by which the trials in the histograms are colored.

colTrial A character vector with plot colors for the trials. A single color when `colorTrialBy = NULL`, a vector of colors otherwise.

trialOrder A character vector indicating the order of the trials in the plot matrix (left to right and top to bottom). This vector should be a permutation of all trials plotted.

addCorr A character string indicating the position of the correlation between trials displayed in each plot, either "tl" for top left, "tr", for top right, "bl" for bottom left or "br" for bottom right. If `NULL`, the default, then no correlation is added to the plot.

See Also

Other functions for TD objects: `TD`, `getMeta()`, `summary.TD()`

Examples

```
data("dropsRaw")

## Create a TD object.
dropsTD <- createTD(data = dropsRaw[dropsRaw$year == 2012, ],
                    genotype = "Variety_ID",
                    trial = "Experiment",
                    loc = "Site",
                    repId = "Replicate",
                    subBlock = "block",
                    rowCoord = "Row",
                    colCoord = "Column",
                    trLat = "Lat",
                    trLong = "Long")

### Layout plot.
```

```
## Plot the layout of one of the trials.
plot(dropsTD,
    trials = "Kar12W")

## Highlight some of the genotypes in the layout.
plot(dropsTD,
     trials = "Kar12W",
     highlight = c("A3", "11430"))

### Map plot.

## Plot the location of the trials on the map.
plot(dropsTD,
     plotType = "map")

### Box plot.

## Create a box plot for grain.yield.
plot(dropsTD,
     plotType = "box",
     traits = "grain.yield")

## Add coloring by scenarioFull to the boxes.
plot(dropsTD,
     plotType = "box",
     traits = "grain.yield",
     colorTrialBy = "scenarioFull")

## Sort the boxes in descending order.
plot(dropsTD,
     plotType = "box",
     traits = "grain.yield",
     orderBy = "descending")

### Correlation plot.

## Plot the correlations between trials for grain.yield.
plot(dropsTD,
     plotType = "cor",
     traits = "grain.yield")

### Scatter plot

## Plot scatter plot for grain.yield.
plot(dropsTD,
    plotType = "scatter",
    traits = "grain.yield")

## Create a scatter plot matrix for grain yield.
## Color trials by scenario and genotypes by family.
plot(dropsTD,
     plotType = "scatter",
     traits = "grain.yield",
```

```
        colorTrialBy = "scenarioFull",
        colorGenoBy = "geneticGroup")
```

`report` *Base method for creating a report*

Description

Base method for creating a .pdf and .tex report from an R object.

Usage

```
report(x, ...)
```

Arguments

x	An R object
...	Further arguments to be passed on to specific report functions.

See Also

report.STA

`report.STA` *Report method for class STA*

Description

pdf reports will be created containing a summary of the results of the fitted model(s). For all selected trails and traits a separate pdf file will be generated. Also a .tex file and a folder containing figures will be created for each report to enable easy modifying of the report.

Usage

```
## S3 method for class 'STA'
report(
  x,
  ...,
  trials = NULL,
  traits = NULL,
  descending = TRUE,
  outfile = NULL,
  what = c("fixed", "random")
)
```

Arguments

`x`	An object of class STA.
`...`	Further arguments passed to the report function.
`trials`	A character vector indicating the trials to report. If `trials = NULL`, all trials are reported.
`traits`	A character vector indicating the traits to report. If `traits = NULL`, all traits are reported.
`descending`	Should the trait be ordered in descending order? Set to `FALSE` if low values of the trait indicate better performance.
`outfile`	A character string, the name and location of the output .pdf and .tex file for the report. If `NULL`, a report with a default name will be created in the current working directory. Trial, trait and the type of model (genotype fixed or random) will be concatenated to the name of the output file. Both knitr and pdflatex don't work well with spaces in file paths and these are therefore disallowed. Relative paths are possible though.
`what`	A character vector indicating whether the fitted model with genotype as fixed or genotype as random factor should be reported. By default all fitted models in the STA object are reported.

Details

This function uses pdflatex to create a pdf report. For it to run correctly an installation of LaTeX is required. Checking for this is done with Sys.which("pdflatex").

Value

A pdf report and the .tex file and figures folder that can be used to recreate the report.

See Also

Other functions for STA objects: `STAtoCross()`, `STAtoTD()`, `plot.STA()`, `summary.STA()`

Examples

```
## Fit model using lme4.
modLme <- fitTD(TD = TDHeat05,
                design = "ibd",
                traits = "yield",
                engine = "lme4")

## Create a pdf report summarizing the results for the model with genotype
## as fixed factor.

report(modLme,
       outfile = tempfile(fileext = ".pdf"),
       what = "fixed")
```

```
## Create two pdf report summarizing the results for the model with genotype
## as fixed factor and for the model with genotype as random factor. Order
## the results in ascending order.

report(modLme,
       outfile = tempfile(fileext = ".pdf"),
       descending = FALSE)
```

STAtoCross *Convert STA to Cross*

Description

Convert an STA object to a cross object from package qtl. Genotypic information should be available in a .csv file.
The only way to create an object of class cross is by importing both the phenotypic and the genotypic data from external files. Therefore the phenotypic data, either the BLUEs or the BLUPs from the fitted model are first written to a temporary file. The genotypic data has to be available in a .csv file in the correct format as well, see `genoFile` for a description of this format. These phenotypic and genotypic files are then imported into a cross object using the read.cross function in the qtl package.

Usage

```
STAtoCross(
  STA,
  trial = NULL,
  traits = NULL,
  what = c("BLUEs", "BLUPs"),
  genoFile,
  genotypes = c("A", "H", "B", "D", "C"),
  ...
)
```

Arguments

`STA`	An object of class `STA`.
`trial`	A character string indicating the trial to be exported. If `NULL` and `STA` contains only one trial, that trial is exported.
`traits`	A character string containing the traits to be exported. If `NULL`, all traits for the selected trial are exported.
`what`	A character string containing the statistics to be exported as phenotype in the cross object. This can be either `BLUEs` or `BLUPs`.

genoFile — A character string indicating a filename containing phenotypic data. The data should be in the format required by the qtl package. The first column should contain the individuals, starting from row 4. The following columns contain markers with in the second and third row the chromosome and position on the chromosome and in the following rows the genotypes.

genotypes — A character vector specifying the genotype codes corresponding to AA, AB, BB, not BB and not AA.

... — Further arguments to be passed to the read.cross function. See read.cross.

See Also

read.cross

Other functions for STA objects: STAtoTD(), plot.STA(), report.STA(), summary.STA()

Examples

```
## Fit model using SpATS.
modSp <- fitTD(TD = TDHeat05,
               design = "res.rowcol",
               traits = "yield",
               what = "fixed")

## Create cross object with BLUEs from modSp using genotypic information
## from markers.csv in the package.
cross <- STAtoCross(modSp,
                    genoFile = system.file("extdata", "markers.csv",
                                           package = "statgenSTA"))
```

STAtoTD *Convert STA to TD*

Description

Convert an STA object to a TD object.
To be able to use the output of a single trial analysis in Genotype-by-Environment (GxE) analysis the output first needs to be converted back to an TD object. This function does exactly that. It extracts BLUEs, BLUPs and their standard errors from the STA object and creates a new TD object using these. Also a column "wt" (weight) may also be added. Weights are then calculated as 1/(SE BLUEs) ^ 2.

Usage

```
STAtoTD(
  STA,
  what = c("BLUEs", "seBLUEs", "BLUPs", "seBLUPs"),
  traits = NULL,
  keep = NULL,
```

```
  addWt = FALSE
)
```

Arguments

STA	An object of class STA.
what	A character string containing the statistics to be included as traits in the TD object. Multiple statistics can be included in which case they will appear as statistic_trait in the output
traits	A character string containing the traits to be included in the TD object. If NULL, all traits are exported.
keep	Columns from the TD object used as input for the STA model to be copied to the output. see extractSTA for possible columns to copy. If if it is available in TD, the column trial will always be copied.
addWt	Should a column wt be added to the output? If TRUE weight is calculated as 1/(SE BLUEs) ^ 2. If multiple traits are included in the output, multiple weight columns will be added, 1 for each trait. These will be named wt_trait.

Details

Trial information for the trials in the STA object will be copied from the original TD object on which the modeling was done.

See Also

Other functions for STA objects: STAtoCross(), plot.STA(), report.STA(), summary.STA()

Examples

```
## Fit model using SpATS.
modSp <- fitTD(TD = TDHeat05,
               design = "res.rowcol",
               traits = "yield",
               what = "fixed")

## Create TD object from the fitted model with BLUEs and standard errors.
TDRes <- STAtoTD(modSp,
                 what = c("BLUEs", "seBLUEs"))

## Add a weight column in the output.
TDResWt <- STAtoTD(modSp,
                   what = c("BLUEs", "seBLUEs"),
                   addWt = TRUE)
```

summary.STA *Summarizing objects of class* STA

Description

summary method for class STA.

Usage

```
## S3 method for class 'STA'
summary(
  object,
  trials = NULL,
  trait = NULL,
  nBest = 20,
  sortBy = NULL,
  naLast = TRUE,
  decreasing = TRUE,
  ...
)
```

Arguments

object	An object of class STA.
trials	A character vector indicating the trial to summarize. If trial = NULL a summary is made of all trials in the STA object. If a single trial is selected a full summary for this trial is created. For multiple trials a summary table with the most importantant statistics is returned.
trait	A character string indicating the trait to summarize. If trait = NULL and only one trait is modeled, this trait is summarized.
nBest	An integer indicating the number of the best genotypes (sorted by either BLUEs or BLUPs) to print. If NA, all genotypes will be printed.
sortBy	A character string specifying how the genotypes will be sorted. Either "BLUEs", "BLUPs" or NA (i.e. no sorting).
naLast	Should missing values in the data be put last when sorting?
decreasing	Should the sort order be decreasing?
...	Further arguments - not used.

See Also

Other functions for STA objects: STAtoCross(), STAtoTD(), plot.STA(), report.STA()

Examples

```
## Run a single trait analysis using SpATS.
modSp <- fitTD(TD = TDHeat05,
               design = "res.rowcol",
               traits = "yield")

## Print a summary of the fitted model.
summary(modSp)
```

---

summary.TD *Summarizing objects of class* TD

Description

summary method for class TD.

Usage

```
## S3 method for class 'TD'
summary(
  object,
  ...,
  trial = names(object),
  traits,
  groupBy = NULL,
  what = if (!is.null(groupBy)) {
     c("nObs", "mean", "sd")
 } else {
     c("nObs",
    "nMiss", "mean", "median", "min", "max", "firstQ", "thirdQ", "var")
 }
)
```

Arguments

object	An object of class TD.
...	Further arguments - currently not used.
trial	A character string specifying the trial to be summarized.
traits	A character vector specifying the traits to be summarized.
groupBy	A character string specifying a column in TD by which the summary should be grouped. If NULL, no grouping is done.
what	A character vector indicating which summary statistics should be computed. If what = "all", all available statistics are computed. Possible options are:

nVals The number of values, i.e. non-missing + missing values.
nObs The number of non-missing observations.
nMiss The number of missing values.
mean The mean.
median The median.
min The minimum.
max The maximum.
range The range (maximum - minimum).
firstQ The first (25pct) quantile.
thirdQ The third (75pct) quantile.
sd The standard deviation.
seMean The standard error of mean.
var The variance.
seVar The standard error of variance.
CV The coefficient of variation.
sum The sum.
sumSq The sum of squares.
uncorSumSq The uncorrected sum of squares.
skew The skewness.
seSkew The standard error of the skewness.
kurt The kurtosis.
seKurt The standard error of the kurtosis.
all All summary statistics.

Value

A table containing the selected summary statistics.

See Also

Other functions for TD objects: `TD`, `getMeta()`, `plot.TD()`

Examples

```
## Summarize TDHeat05.
summary(TDHeat05,
        traits = "yield")

## Summarize TDHeat05 grouping by repId.
summary(TDHeat05,
        traits = "yield",
        groupBy = "repId")
```

TD *S3 class TD*

Description

`createTD`
Function for creating objects of S3 class TD (Trial Data). The function converts a data.frame to an object of class TD in the following steps:

- Check input data
- Rename columns to default column names - default column names: genotype, trial, loc, year, repId, subBlock, rowCoord, colCoord, rowId, colId, checkId
- Convert column types to default column types - rowCoord and colCoord are converted to numeric columns, all other renamed columns to factor columns. Columns other than the default columns, e.g. traits or other covariates will be included in the output unchanged.
- Split input data by trial - each trial in the input data will become a list item in the output.
- Add meta data - the trial meta data are added as attributes to the different output items. The function parameters starting with "tr" provide the meta data. Their values will be recycled if needed, so by setting a single "trDesign", all trials will get the same design. For trLat, trLong, trDesign and trDate a column in `data` that contains the information can be specified as well. The meta data can be changed later on using `getMeta` and `setMeta`

`addTD`
Function for adding extra trial data to an existing object of class TD. The data for the new trials will be added after the data for existing trials. It is possible to add data for an already existing trial, but this will cause multiple items in the output with identical names, which might cause problems later on in the analysis. Therefore a warning will be issued in this case.

`dropTD`
Function for removing data for selected trials from an existing object of class TD.

`summary.TD` and `plot.TD` methods are available.

Usage

```
createTD(
  data,
  genotype = NULL,
  trial = NULL,
  loc = NULL,
  year = NULL,
  repId = NULL,
  subBlock = NULL,
  plotId = NULL,
  rowCoord = NULL,
  colCoord = NULL,
```

```
  rowId = rowCoord,
  colId = colCoord,
  checkId = NULL,
  trLocation = NULL,
  trDate = NULL,
  trDesign = NULL,
  trLat = NULL,
  trLong = NULL,
  trPlWidth = NULL,
  trPlLength = NULL
)

addTD(
  TD,
  data,
  genotype = NULL,
  trial = NULL,
  loc = NULL,
  year = NULL,
  repId = NULL,
  subBlock = NULL,
  plotId = NULL,
  rowCoord = NULL,
  colCoord = NULL,
  rowId = rowCoord,
  colId = colCoord,
  checkId = NULL,
  trLocation = NULL,
  trDate = NULL,
  trDesign = NULL,
  trLat = NULL,
  trLong = NULL,
  trPlWidth = NULL,
  trPlLength = NULL
)

dropTD(TD, rmTrials)
```

Arguments

`data`	A data.frame containing trial data with at least a column for genotype. The data.frame should be in a wide format, i.e. all available phenotypic data should be in a separate column within the data.frame.
`genotype`	An optional character string indicating the column in `data` that contains genotypes.
`trial`	An optional character string indicating the column in `data` that contains trials.
`loc`	An optional character string indicating the column in `data` that contains trial locations.

`year`	An optional character string indicating the column in `data` that contains years.
`repId`	An optional character string indicating the column in `data` that contains replicates.
`subBlock`	An optional character string indicating the column in `data` that contains sub blocks.
`plotId`	An optional character string indicating the column in `data` that contains plots. This column will be combined with trial to a single output factor.
`rowCoord`	An optional character string indicating the column in `data` that contains the row coordinates.
`colCoord`	An optional character string indicating the column in `data` that contains the column coordinates.
`rowId`	An optional character string indicating the column in `data` that contains field rows. If not supplied, this is assumed to be the same as rowCoord.
`colId`	An optional character string indicating the column in `data` that contains field columns. If not supplied, this is assumed to be the same as colCoord.
`checkId`	An optional character string indicating the column in `data` that contains the check IDs.
`trLocation`	An optional character vector indicating the locations of the trials. This will be used as default names when creating plots and summaries. If no locations are provided, first the column loc is considered. If this contains one unique value for a trial this is used as trLocation. Otherwise the trial name is used.
`trDate`	An optional character string indicating the column in `data` that contains the date of the trial or a date vector indicating the dates of the trials.
`trDesign`	An optional character string indicating the column in `data` that contains the design of the trial or a character vector indicating the designs of the trials. Either "none" (no (known) design), " ibd" (incomplete-block design), "res.ibd" (resolvable incomplete-block design), "rcbd" (randomized complete block design), "rowcol" (row-column design) or "res.rowcol" (resolvable row-column design).
`trLat`	An optional character string indicating the column in `data` that contains the latitude of the trial or a numerical vector indicating the latitudes of the trials on a scale of -90 to 90.
`trLong`	An optional character string indicating the column in `data` that contains the latitude of the trial or a numerical vector indicating the longitudes of the trials on a scale of -180 to 180.
`trPlWidth`	An optional positive numerical vector indicating the widths of the plots.
`trPlLength`	An optional positive numerical vector indicating the lengths of the plots.
`TD`	An object of class TD which should be modified.
`rmTrials`	A character vector of trials that should be removed.

Value

An object of class TD, a list of data.frames with renamed columns and an attribute `renamedCols` containing an overview of renamed columns. For each unique value of trial, the output has a data.frame in the list with the same name as the trial. These data.frames have attributes containing the metadata for the corresponding trial. If there is no column for trial, the list will contain one item named after the input data.

See Also

Other functions for TD objects: getMeta(), plot.TD(), summary.TD()

Examples

```
## Create a data.frame to be converted to TD object.
## The data consists of genotype, trial, row and column information and
## two traits, yield and flowering time.
datT1 <- data.frame(geno = paste0("G", 1:10),
                    tr = "T1",
                    row = rep(1:5, each = 2),
                    col = rep(1:2, times = 5),
                    yield = 1:10,
                    flowering = 3:12)

## Convert data.frame to TD object.
TDT1 <- createTD(data = datT1,
                 genotype = "geno",
                 trial = "tr",
                 rowCoord = "row",
                 colCoord = "col")

## Create a second data.frame similar to the first with data for a second trial.
datT2 <- data.frame(geno = paste0("G", 1:10),
                    tr = "T2",
                    row = rep(1:2, each = 5),
                    col = rep(1:5, times = 2),
                    yield = 10:1,
                    flowering = 12:3)

## Add this data to the TD object created above.
TDTot <- addTD(TD = TDT1,
               data = datT2,
               genotype = "geno",
               trial = "tr",
               rowCoord = "row",
               colCoord = "col")

## Drop the data for the first trial from the object.
TDT2 <- dropTD(TD = TDTot,
               rmTrials = "T1")
```

TDHeat05 *Field data for a wheat experiment in Mexico*

Description

A dataset converted to a TD object containing raw plot data for one trial from a series of wheat trials conducted in Mexico by CIMMYT. The different trials took place under different regimes

of irrigation and temperature, there were 4 trials across two years, labelled as DRIP05, HEAT05, HEAT06 and IRRI06. The TD object only contains the data for HEAT05. Within each trial, a set of 167 progeny of a RIL (Recombinant Inbred Line; 8 generations) population were tested alongside the population parents (Seri and Babax). A lattice design with two replicates was used for each trial. In the first replicate the entries were not randomized, as they were considered to be a random selection from a population.

Usage

```
TDHeat05
```

Format

A TD object, a list containing a data.frame with the following columns:

trial trial, a combination of watering regime, year and nitrogen treatment

genotype genotype

Plot plot number in the field

repId replicate

subBlock block id

rowId row within the field (as factor)

colId column within the field (as factor)

yield yield in grams per square meter

rowCoord row within the field (as numerical value)

colCoord column within the field (as numerical value)

`TDMaize` *Field data for a maize experiment in Tlaltizapan, Mexico*

Description

A dataset converted into a TD object containing data corresponding to an F2 maize reference population from CIMMYT maize drought breeding program, which was derived from the cross of a drought-tolerant line (P1) with a drought susceptible line (P2) as described in detail by Ribaut et al. (1996, 1997).
DNA from 211 F2 plants was extracted to produce information for 132 co-dominant markers on 10 linkage groups. Phenotypic evaluations were performed on 211 F2:3 families, each one derived from an original F2 plant. The families were evaluated under different water and nitrogen regimes during 1992, 1994 and 1996. In the winter of 1992 three water regimes were imposed on the trials: well watered (NS), intermediate stress (IS) and severe stress (SS). In the winter of 1994, only the IS and SS trials were available. Nitrogen availability varied in the 1996 trials, with two low nitrogen treatments (LN, in winter and summer) and one high-nitrogen treatment (HN in summer). In each of the trials, five traits were evaluated but only grain yield is included in the data.

Usage

```
TDMaize
```

Format

A TD object, a list containing 8 data.frames, each with the following columns:

trial trial, a combination of watering regime, year and nitrogen treatment

genotype genotype

regime stress level, NS (no water stress), IS (intermediate water stress, SS (severe water stress), LN (low nitrogen) or HN (high nitrogen))

yld grain yield in tons

Source

https://link.springer.com/article/10.1007/BF00221905

References

Ribaut JM, Hoisington DA, Deutsch JA, Jiang C, Gonzalez de Leon D (1996) Identification of quantitative trait loci under drought conditions in tropical maize.1. Flowering parameters and the anthesis-silking interval. Theor Appl Genet 92:905–914

Ribaut JM, Jiang C, Gonzalez de Leon D, Edmeades GO, Hoisington DA (1997) Identification of quantitative trait loci under drought conditions in tropical maize.2. Yield components and marker-assisted selection strategies. Theor Appl Genet 94:887–896

Index

∗ **datasets**
- dropsRaw, 2
- TDHeat05, 31
- TDMaize, 32

∗ **functions for STA objects**
- plot.STA, 14
- report.STA, 20
- STAtoCross, 22
- STAtoTD, 23
- summary.STA, 25

∗ **functions for TD objects**
- getMeta, 10
- plot.TD, 16
- summary.TD, 26
- TD, 28

addTD *(TD)*, 28

createTD *(TD)*, 28

dropsRaw, 2
dropTD *(TD)*, 28

extractSTA, 4, *24*

fitTD, *6*, 6, *9*

getMeta, 10, *18*, *27*, *31*

outlierSTA, 12

plot.STA, 14, *21*, *23–25*
plot.TD, *12*, 16, *27*, *28*, *31*
PSANOVA, *7*, *8*

read.cross, *23*
report, 20
report.STA, *15*, *20*, 20, *23–25*

setMeta, *7*
setMeta *(getMeta)*, 10

STAtoCross, *15*, *21*, 22, *24*, *25*
STAtoTD, *15*, *21*, *23*, 23, *25*
summary.STA, *15*, *21*, *23*, *24*, 25
summary.TD, *12*, *18*, 26, *28*, *31*

TD, *5*, *7*, *9*, *12*, *18*, *27*, 28
TDHeat05, 31
TDMaize, 32