# Package ‘MMGS’

July 14, 2025

**Type** Package

**Title** Multi-Methods for Multi-Envs Genomic Selection Pipeline

**Version** 1.0.0

**Maintainer** Eason Zhu <z980907mj@gmail.com>

**Description** Package for Multi-
envs Genomic Selection, contains Inbred Lines, CUBIC Lines and Light-cross Lines.

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**Imports** BGLR, e1071, randomForest, corrgram, rrBLUP, lemon, glmnet,
lightgbm, tidyr, stringi, reshape2, grDevices, dplyr, stats,
colorspace

**Depends** ggplot2, R (>= 2.10)

**LazyData** true

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Eason Zhu [aut, cre]

## R topics documented:

## `envMeanPara` *envMeanPara*

### Description

Calculate environmental mean parameters for a specific trait within a specified time range. This time range is calculated by the Exhaustive_search.

### Usage

```
envMeanPara(data, env_paras,
 maxR_dap1 = NULL, maxR_dap2 = NULL,
 Paras = NULL)
```

### Arguments

| | |
|---|---|
| `data` | Data frame containing trait mean values within different environments. |
| `env_paras` | Data frame containing environmental parameter information. |
| `maxR_dap1` | The starting day for calculating environmental mean parameters. Default is 18, which is used for the test data. |
| `maxR_dap2` | The ending day for calculating environmental mean parameters. Default is 43, which is used for the test data. |
| `Paras` | Vector containing the names of environmental parameters to be calculated. |

### Value

A data frame containing trait mean values and calculated environmental mean parameters.

### Examples

```
## Not run:
envMeanPara<-envMeanPara(data=env_trait,env_paras=PTT_PTR)

## End(Not run)
```

`envMeanPara_plotter` *envMeanPara_plotter*

## Description

Generate a plot of environmental parameter means against the overall mean.

## Usage

```
envMeanPara_plotter(
  data,
  size = NULL,
  shape = NULL,
  method = NULL,
  Paras = NULL,
  env_code = NULL,
  linewidth = NULL,
  alpha = NULL,
  linetype = NULL,
  linecolor = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | An environmental parameter mean data frame. |
| `size` | Point size for scatter plot. |
| `shape` | Point shape for scatter plot. |
| `method` | Regression method for the smoothing line. |
| `Paras` | Vector containing the names of environmental parameters to be considered. |
| `env_code` | Point color for the scatter plot by env_code.Defualt is env_code. |
| `linewidth` | Line width for the smoothing line. |
| `alpha` | Alpha value for points in the scatter plot. |
| `linetype` | Line type for the smoothing line. |
| `linecolor` | Line color for the smoothing line. |

## Value

A plotter displaying the relationship between environmental parameter means and the overall mean.

## Examples

```
## Not run: envMeanPara_plotter(envMeanPara,
Paras=c('DL', 'GDD', 'PTT', 'PTR', 'PTS'))
## End(Not run)
```

## env_info *env_info Dataset*

### Description

This dataset contains information about environmental factors for various environments.

### Usage

```
data(env_info)
```

### Format

A data frame with the following columns:

**env_code** Environmental code assigned to each environment.

**env_notes** Environmental note assigned to each environment.

**lat** Latitude of the environment.

**lon** Longitude of the environment.

**Location** Location of the environment.

**PlantingDate** Planting Date of lines.

**TrialYear** Planting Year of lines.

### Source

This dataset originates from the article and provides essential information for environmental analysis.

### Examples

```
data(env_info)
```

## env_trait_calculate *env_trait_calculate*

### Description

Calculate the mean and optional quantiles for a trait across different environments.

### Usage

```
env_trait_calculate(data, trait, env, q25_75 = TRUE)
```

### Arguments

| | |
|---|---|
| `data` | Input data frame containing trait values and environmental information. |
| `trait` | The name of the trait for which mean and optional quantiles will be calculated. |
| `env` | The column name in the data frame representing different environments. |
| `q25_75` | Logical, indicating whether to calculate the 25th and 75th quantiles. Default is TRUE |

### Value

Data frame containing the mean trait values for each environment. If q25_75 is TRUE, it also includes columns for the 25th(q25), 75th(q75) quantiles, as well as the number of observations (n_obs).

### Examples

```
env_trait<-env_trait_calculate(data=trait,trait="FTgdd",env="env_code")

# Access the result, e.g., mean trait values
env_trait$mean

# Access the result, e.g., 25th quantile values
env_trait$q25

# Access the result, e.g., 75th quantile values
env_trait$q75

# Access the result, e.g., number of observations
env_trait$n_obs
```

## `etl_plotter` *etl_plotter*

### Description

Generate a plot of environmental trait lines with mean, q25, and q75 areas.

### Usage

```
etl_plotter(
  data,
  trait,
  env_cols = NULL,
  shape = NULL,
  size = NULL,
  linewidth = NULL,
  area_color = NULL,
  area_alpha = NULL
)
```

### Arguments

| | |
|---|---|
| `data` | An etl input data frame. |
| `trait` | An env_trait data frame containing mean, q25, and q75 values. |
| `env_cols` | Environmental colors gradient; if NULL, it generates colors using rainbow_hcl. |
| `shape` | Point shape, similar to ggplot2. |
| `size` | Point size, similar to ggplot2. |
| `linewidth` | Line width, similar to ggplot2. |
| `area_color` | Fill color for the area between q25 and q75. |
| `area_alpha` | Fill color alpha for the area between q25 and q75. |

### Value

A plot of etl, filling the area between env-trait q25 to q75, with a red line representing the env-trait mean.

### Examples

```
env_trait<-env_trait_calculate(data=trait,trait="FTgdd",env="env_code")
LbyE<-LbyE_calculate(trait,trait="FTgdd",env="env_code",line="line_code")
etl<-LbyE_Reshape(data=env_trait,LbyE=LbyE,env="env_code")
#Plot
etl_plotter(data=etl,trait=env_trait)
```

## `Exhaustive_plotter` *Exhaustive_plotter*

### Description

Generate a combined plot for pairwise correlations using pop_cor data.

### Usage

```
Exhaustive_plotter(Correlation,
 dap_x, dap_y, p = NULL,
 Paras, Cor = NULL)
```

### Arguments

| | |
|---|---|
| `Correlation` | A pop_cor data frame containing correlation values for different pairs. |
| `dap_x` | Specifies the column for the x-axis variable. |
| `dap_y` | Specifies the column for the y-axis variable. |
| `p` | A parameter with a default value of 1. |
| `Paras` | Vector containing the names of environmental parameters to be considered. |
| `Cor` | Assigning variable column names.Default is Cor. |

### Value

A combined pop_cor plot.

### Examples

```
## Not run: Exhaustive_plotter(Correlation,dap_x,dap_y,Paras)
```

---

## `Exhaustive_search` *Exhaustive_search*

---

### Description

Perform exhaustive search for population correlation matrix based on environmental parameters.

### Usage

```
Exhaustive_search(
  data,
  env_paras,
  searching_daps,
  p = NULL,
  dap_x = NULL,
  dap_y = NULL,
  LOO = NULL,
  Paras = NULL
)
```

### Arguments

| | |
|---|---|
| `data` | Data frame containing environmental trait data. |
| `env_paras` | Data frame containing environmental parameter information. |
| `searching_daps` | The total number of days to search for, typically set based on your data. Default is 122. |
| `p` | The parameter for controlling the shape of the search window. Default is 1. |
| `dap_x` | The number of days for the x-axis of the search window. Default is the same as searching_daps. |
| `dap_y` | The number of days for the y-axis of the search window. Default is the same as searching_daps. |
| `LOO` | Leave-One-Out cross-validation flag. If LOO is 1, |
| `Paras` | Vector containing the names of environmental parameters to be considered. |

### Value

A population correlation matrix based on the exhaustive search.

### Examples

```
#Input from function : env_trait_calculate
env_trait<-env_trait_calculate(data=trait,trait="FTgdd",env="env_code")
#Run
## Not run: Exhaustive_search(denv_trait,PTT_PTR)
```

## geno *geno Dataset*

### Description

This dataset contains genetic information for various lines.

### Usage

```
data(geno)
```

### Format

An object of class `data.frame` with 237 rows and 1463 columns.

### Source

This dataset originates from the article and provides genetic data for analysis.

### Examples

```
data(geno)
```

## LbyE_calculate *LbyE_calculate*

### Description

Calculate the lines in all environments.

### Usage

```
LbyE_calculate(data, trait, env, line)
```

### Arguments

| | |
|---|---|
| `data` | Input data frame containing trait values, line information, and environmental information. |
| `trait` | The trait for which the lines within different environments will be calculated. |
| `env` | Column name representing different environments in the input data frame. |
| `line` | Column name representing different lines in the input data frame. |

### Value

A data frame containing line values within different environments for the specified trait.

### Examples

```
# Calculate lines within different environments for the trait "FTgdd"
LbyE_calculate(trait,"FTgdd","env_code","line_code")
```

## LbyE_corrplot *LbyE_corrplot*

### Description

Plot the correlation between lines and environments.

### Usage

```
LbyE_corrplot(LbyE, cor_type = NULL, color = NULL)
```

### Arguments

| | |
|---|---|
| LbyE | A data frame calculated by LbyE_calculate. Column names represent Lines, and row names represent Environments. |
| cor_type | Graphic format of the correlation map. The default format is "heatmap". Other option is "pie". |
| color | Set the color gradient for the correlation map. It is a vector of three colors representing low, mid, and high values. |

### Value

A correlation plot between lines and environments in the specified format.

### Examples

```
LbyE<-LbyE_calculate(trait,"FTgdd","env_code","line_code")
# Generate a heatmap correlation plot for Lines and Environments
colors=c("blue","white","red")
LbyE_corrplot(LbyE,cor_type="heatmap",color=colors)
# Generate a pie chart correlation plot for Lines and Environments
LbyE_corrplot(LbyE,cor_type="pie",color=colors)
```

## LbyE_Reshape *LbyE_Reshape*

### Description

Reshape trait values within specified environments and lines.

### Usage

```
LbyE_Reshape(data, LbyE, env)
```

### Arguments

| | |
|---|---|
| data | Data frame containing environmental information from env_trait_calculate. |
| LbyE | A data frame calculated by LbyE_calculate. Column names represent Lines,and row names represent Environments. |
| env | Column name representing different environments in the data frame. |

### Value

Data frame containing trait values within specified environments and lines.

### Examples

```
env_trait<-env_trait_calculate(trait,"FTgdd","env_code")
LbyE<-LbyE_calculate(trait,"FTgdd","env_code","line_code")
#Run:
etl<-LbyE_Reshape(env_trait,LbyE,"env_code")
```

## line_trait_mean *line_trait_mean*

### Description

Calculate the mean of residuals for each line after fitting a linear model.

### Usage

```
line_trait_mean(data, LbyE, mean, trait, row)
```

### Arguments

| | |
|---|---|
| data | Input data frame containing trait values, line information, and environmental information. |
| LbyE | Data frame calculated by LbyE_calculate, containing line values within different environments. |
| mean | Data frame containing mean trait values within different environments. This data frame get from the function of env_mean_calculate. |
| trait | The trait for which the residuals will be calculated. |
| row | The minimum number of rows required for fitting a linear model for each line. |

### Value

A list containing two elements:

1. data: Data frame with columns env_code, errors (mean of residuals), and additional columns from env_trait.
2. lm_residuals: Data frame containing the residuals squared for each line and environment.

### Examples

```
LbyE<-LbyE_calculate(trait,trait="FTgdd",env="env_code",line="line_code")
env_trait<-env_trait_calculate(data=trait,trait="FTgdd",env="env_code")
#Run
result<-line_trait_mean(data=trait,LbyE=LbyE,
                        mean = env_trait,trait="FTgdd",row=2)
print(result[[1]])
```

## `Mean_trait_plot` *Mean_trait_plot*

### Description

Generate a plot displaying the relationship between the mean trait values and the environmental parameter mean.

### Usage

```
Mean_trait_plot(
  Reg,
  MSE,
  point_size = NULL,
  color = NULL,
  point_shape = NULL,
  linewidth = NULL,
  linetype = NULL
)
```

### Arguments

| | |
|---|---|
| `Reg` | Regression result data frame. |
| `MSE` | Mean Squared Error (MSE) result data frame. |
| `point_size` | Size of points in the plot. |
| `color` | color for the points and the dash area. |
| `point_shape` | Shape of points in the plot. |
| `linewidth` | Line width for the smoothing line. |
| `linetype` | Line type for the smoothing line. |

## Value

A plot showing the relationship between mean trait values and the environmental parameter mean, with MSE information.

## Examples

```
## Not run: Mean_trait_plot(Reg, MSE)
```

---

MMGP *Cross Vadidation and Genomic Prediction Cross environments*

---

## Description

MMGP (Multi-environment Multi-Genomic Prediction) is a function for genomic prediction that incorporates cross-validation and cross-environment prediction. It utilizes both genetic and environmental information to predict phenotypic values. The function supports various genomic prediction methods and models, allowing users to choose the most suitable approach based on data characteristics and preferences.

## Usage

```
MMGP(
  pheno,
  geno,
  env,
  para,
  Para_Name,
  model,
  depend = NULL,
  fold = NULL,
  reshuffle = NULL,
  methods = NULL,
  ENalpha = NULL,
  SVM_cost = NULL,
  gamma = NULL,
  kernel = NULL,
  nIter = NULL,
  burnIn = NULL,
  thin = NULL,
  GBM_params = NULL,
  GBM_rounds = NULL,
  Envs = NULL,
  Obs = NULL,
  line_code = NULL,
  ms1 = NULL,
  ms2 = NULL
)
```

### Arguments

`pheno` Vector (n x j) of "phenotypes," i.e., observations or pre-processed, corrected values.

`geno` Matrix (n x m) of genotypes for the training population: n lines with m markers.Genotypes should be coded -1, 0, 1. Missing data are not allowed.

`env` Data.frame (j x l) of "environmental information," i.e., environmental conditions corresponding to phenotypes.

`para` Data.frame returned from envMeanPara function, providing information about environmental means and parameters.

`Para_Name` The most relevant environmental covariates to the subject's phenotype, obtained after stepwise correlation calculations.

`model` The options for genomic breeding cross vadiation methods.

`depend` The options for genomic breeding within different environment. The available options are: 1.Norm(Reaction Norm Model) 2.PEI(Polygenic Environment Interaction Model)

`fold` Number of folds for cross-validation. Smallest recommended value is fold = 2.

`reshuffle` Number of independent replicates for cross-validation. Smallest recommended value is reshuffle = 5.

`methods` A vector specifying the prediction methods to be used.

`ENalpha` Used for Elastic Net prediction model, with a value range from 0 to 1.

`SVM_cost` Cost of constraints violation for SVM (default: 10).

`gamma` Parameter needed for all kernels except linear (default: 0.001).

`kernel` the kernel used in training and predicting. You might consider changing some of the following parameters, depending on the kernel type. linear, polynomial, radial basis, sigmoid.

`nIter` Number of iterations for the Bayesian model.

`burnIn` Number of burn-in iterations for the Bayesian model.

`thin` Thinning parameter for the Bayesian model.

`GBM_params` A list of parameters for LightGBM. See LightGBM documentation for details.

`GBM_rounds` Number of training rounds for LightGBM.

`Envs` Assign variable column names of Envs. Default is Envs.

`Obs` Assign variable names of Observed values. Default is Obs.

`line_code` Assign variable column names of line_code.Default is line_code.

`ms1` Remove the line for which the number of missing environments > ms1.

`ms2` Remove the line for which the number of missing environments > ms2.

### Value

The class MMGP returns a list containing: The Observed and Predicted Values of each line in Multiple Environments. The Predictive accuracy in Multiple Environments.Calculated by Pearson Cor. The Predictive accuracy for All Environments.(Cross R2)

### Examples

```
## Not run:
out<-MMGP(pheno=pheno,geno=geno,env=env_info,
          para=envMeanPara,Para_Name="PTT",
          depend="Norm",fold=2,reshuffle=5,methods="RM.G",
          ms1=ms1,ms2=ms2)

## End(Not run)
```

## MMPrdM *MMPrdM*

### Description

MMPrdM (Multi-environment Multi-marker Prediction Model) is a function for genomic prediction that includes cross-environment prediction. It leverages genetic and environmental information to predict phenotype. The function supports various genomic prediction methods and models, providing flexibility in choosing the most suitable approach based on the data characteristics and user preferences.

### Usage

```
MMPrdM(
  pheno,
  geno,
  env,
  para,
  Para_Name,
  model,
  depend = NULL,
  reshuffle = NULL,
  methods = NULL,
  ENalpha = NULL,
  SVM_cost = NULL,
  gamma = NULL,
  kernel = NULL,
  Envs = NULL,
  line_code = NULL,
  nIter = NULL,
  burnIn = NULL,
  thin = NULL,
  GBM_params = NULL,
  GBM_rounds = NULL
)
```

### Arguments

| | |
|---|---|
| pheno | Vector (n x j) of "phenotypes", i.e. observations or pre-processed, corrected values. |
| geno | Matrix (n x m) of genotypes for the training population: n lines with m markers. Genotypes should be coded -1, 0, 1. Missing data are not allowed. |

`env` Data.frame (j x l) of "environmental information", i.e.

`para` Data.frame returned from envMeanPara function.

`Para_Name` The most relevant environmental silvers to the subject's phenotype, obtained after stepwise correlation calculations, are referred to for more details:

`model` The options for genomic breeding cross vadiation methods. The available options are: 1.GBLUP: performs G-BLUP using a marker-based relationship matrix, implemented through BGLR R-library. Equivalent to ridge regression (RR-BLUP) of marker effects. 2.RR: ridge regression, using package glmnet. In theory, strictly equivalent to gblup. 3.LASSO: Least Absolute Shrinkage and Selection Operator is another penalized regression methods which yield more shrinked estimates than RR.Run by glmnet library. 4.EN: Elastic Net (Zou and Hastie, 2005), which is a weighted combination of RR and LASSO, using glmnet library 5.rrBLUP:the RR-BLUP mixed model(Endelman ,2011). One application is to estimate marker effects by ridge regression; alternatively, BLUPs can be calculated based on an additive relationship matrix or a Gaussian kernel. Several Bayesian methods, using the BGLR library: 1.BRR: Bayesian ridge regression: same as rr-blup, but bayesian resolution. Induces homogeneous shrinkage of all markers effects towards zero with Gaussian distribution (de los Campos et al, 2013) 2.BL: Bayesian LASSO: uses an exponential prior on marker variances priors, leading to double exponential distribution of marker effects (Park & Casella 2008) 3.BA: uses a scaled-t prior distribution of marker effects. (Meuwissen et al 2001). 4.BB: Bayes B, uses a mixture of distribution with a point mass at zero and with a slab of non-zero marker effects with a scaled-t distribution (Habier et al 2011). 5.BC: Bayes C same as Bayes B with a slab with Gaussian distribution. A more detailed description of these methods can be found in Perez & de los Campos 2014 (http://genomics.cimmyt.org/BGLR-extdoc.pdf). Four semi-parametric methods: 1.RKHS: reproductive kernel Hilbert space and multiple kernel MRKHS, using BGLR (Gianola and van Kaam 2008). Based on genetic distance and a kernel function to regulate the distribution of marker effects. This methods is claimed to be effective for detecting non additive effects. 2.RF: Random forest regression, using randomForest library (Breiman, 2001, Breiman and Cutler 2013). This methods uses regression models on tree nodes which are rooted in bootstrapping data. Supposed to be able to capture interactions between markers 3.SVM: support vector machine, run by e1071 library. For details, see Chang, Chih-Chung and Lin, Chih-Jen: LIBSVM: a library for Support Vector Machines http://www.csie.ntu.edu.tw/~cjlin/libsvm 4.LightGBM: Light Gradient Boosting Machine, run by LightGBM library developed by Microsoft. For details, see Jun Yan, Yuetong Xu, at al: LightGBM: accelerated genomically designed crop breeding through ensemble learning

`depend` The options for genomic breeding within different environment.The available options are: 1.Reaction Norm: 2.PEI:

`reshuffle` Number of independent replicates for the Predict. Smallest value recommended is reshuffle = 3.

`methods` RM.G RM.GE RM.E

`ENalpha` used for EN predict model,the value range from 0 to 1.

`SVM_cost` Cost of constraints violation for SVM (default: 10).

`gamma` Parameter needed for all kernels except linear (default: 0.001).

`kernel` the kernel used in training and predicting. You might consider changing some of the following parameters, depending on the kernel type. linear, polynomial, radial basis, sigmoid.

| | |
|---|---|
| `Envs` | Assign variable column names of Envs. Default is Envs. |
| `line_code` | Assign variable column names of line_code Default is line_Code. |
| `nIter` | Number of iterations for the Bayesian model. |
| `burnIn` | Number of burn-in for the Bayesian model. |
| `thin` | Number of thins for the Bayesian model. |
| `GBM_params` | A list of parameters for LightGBM. See LightGBM documentation for details. |
| `GBM_rounds` | Number of training rounds for LightGBM. |

### Value

The class MMPrdM returns a list containing: The Predicted Values of each line in Multiple Environments.

### Examples

```
## Not run:
out<-MMPrdM(pheno=pheno,geno=geno,env=env_info,para=envMeanPara)

## End(Not run)
```

## `mse_plotter` *mse_plotter*

### Description

Generate a plot for Mean Squared Errors (MSE) data analyzed by line_trait_mean.

### Usage

```
mse_plotter(
 data,
 point_size = NULL,
 text_size = NULL,
 point_shape = NULL)
```

### Arguments

| | |
|---|---|
| `data` | Dataframe containing MSE data analyzed by line_trait_mean. |
| `point_size` | Size of points in the plot. |
| `text_size` | Size of text labels in the plot. |
| `point_shape` | Shape of the points in the plot. |

### Value

A plot illustrating Mean Squared Errors (MSE) data.

### Examples

```
## Not run: mse_plotter(MSE)
```

## PTT_PTR *PTT_PTR Dataset*

### Description

Dataset contains information about environmental parameters related to PTT (Photothermal Time) and PTR (Photothermal Ratio) and others.

### Usage

```
data(PTT_PTR)
```

### Format

A data frame representing environmental parameters.

**Date** Date of the environmental information.

**env_code** Environmental code assigned to each environment.

**PTS** Environmental parameter related to Photothermal spectroscopy.

**PTT** Environmental parameter related to Photothermal Time.

**PTR** Environmental parameter related to Photothermal Ratio.

**GDD** Environmental parameter related to Growing Degree Days.

**Tmax** Environmental parameter related to the maximum temperature.

**Tmin** Environmental parameter related to the minimum temperature.

**DL** Environmental parameter related to Day Length.

### Source

This dataset originates from the article and provides environmental parameters for analysis.

### Examples

```
data(PTT_PTR)
```

## Reg *Regression*

### Description

Calculate regression results between traits and environmental means for each line.

### Usage

```
Reg(LbyE, env_trait, filter_num = NULL)
```

### Arguments

| | |
|---|---|
| `LbyE` | Data frame containing lines and their corresponding trait values within environments.It is from the function of LbyE_calculate. |
| `env_trait` | Data frame containing environmental means for different traits within environments.It is from the function of enV_mean_calculate. |
| `filter_num` | Minimum number of observations required to perform regression. Default is 4. |

### Value

A data frame containing regression results for each line.

### Examples

```
#Get Input
LbyE<-LbyE_calculate(trait,trait="FTgdd",env="env_code",line="line_code")
env_trait<-env_trait_calculate(data=trait,trait="FTgdd",env="env_code")
#Calculate
Regression<-Reg(LbyE=LbyE,env_trait=env_trait)
print(head(Regression))
```

## `Reg_plotter` *Reg_plotter*

### Description

Generate a regression plot between the mean and trait, showing the relationship between these two variables.

### Usage

```
Reg_plotter(
 Reg = Reg,
 size = NULL,
 method = NULL,
 color = NULL,
 alpha = NULL)
```

### Arguments

| | |
|---|---|
| `Reg` | Input data frame containing regression results. param x Variable to be plotted on the x-axis. param y Variable to be plotted on the y-axis. param group Variable used for grouping the data. |
| `size` | Size of points in the plot. |
| `method` | Smoothing method for the regression line. |
| `color` | Vector containing colors for regression line and points. |
| `alpha` | Vector containing alpha values for regression line and points. |

### Value

A plot for the regression relationship between mean and trait.

### Examples

```
#' #Get Input
LbyE<-LbyE_calculate(trait,"FTgdd","env_code","line_code")
env_trait<-env_trait_calculate(trait,"FTgdd","env_code")
Regression<-Reg(LbyE = LbyE, env_trait = env_trait)
Reg_plotter(Reg = Regression)
```

## trait *trait Dataset*

### Description

This dataset contains information about phenotypic traits for various lines and environments.

### Usage

```
data(trait)
```

### Format

A data frame representing trait parameters.

**line_code** Line code assigned to each line.

**env_code** Environmental code assigned to each environment.

**env_note** Environmental note assigned to each environment.

**FTdap** Name of the phenotypic trait measured, related to FT.

**FTgdd** Name of the phenotypic trait measured, related to FT.

**pop_code** Other columns present additional trait-related information

### Source

This dataset originates from the article and provides trait data for analysis.

### Examples

```
data(trait)
```