Package 'selection.index'

March 1, 2026

Type Package

Title Analysis of Selection Index in Plant Breeding

Version 2.0.0

Description Provides tools for the simultaneous improvement of multiple traits in plant breeding. Building upon the classical selection index (Smith 1937 <doi:10.1111/j.1469-1809.1936.tb02143.x>) and modern quantitative genetics (Kang 2020 <doi:10.1007/978-3-319-91223-3>), this package calculates classical phenotypic, genomic, marker-assisted, restricted/constrained, and eigen selection indices. It also incorporates multi-stage selection evaluation and stochastic simulations to estimate genetic advance based on economic weights, heritability, and genetic correlations.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Language en-US

Depends R (>= 3.5.0)

Imports MASS, Rcpp, stats, utils

LinkingTo Rcpp, RcppEigen

URL https://github.com/zankrut20/selection.index, https://zankrut20.github.io/selection.index/

BugReports https://github.com/zankrut20/selection.index/issues

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

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation yes

Author Zankrut Goyani [aut, cre, cph]

Maintainer Zankrut Goyani <zankrut20@gmail.com>

Repository CRAN

Date/Publication 2026-02-28 23:10:08 UTC

Contents

`base_index` *Base Index (Williams, 1962)*

Description

Implements the Base Index where coefficients are set equal to economic weights. This is a simple, non-optimized approach that serves as a baseline comparison.

Unlike the Smith-Hazel index which requires matrix inversion, the Base Index is computationally trivial and robust when covariance estimates are unreliable.

Usage

```
base_index(
  pmat,
  gmat,
  wmat,
  wcol = 1,
  selection_intensity = 2.063,
```

```
  compare_to_lpsi = TRUE,
  GAY = NULL
)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits)
gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
wmat	Economic weights matrix (n_traits x k), or vector
wcol	Weight column to use if wmat has multiple columns (default: 1)
selection_intensity	Selection intensity constant (default: 2.063)
compare_to_lpsi	Logical. If TRUE, compares Base Index efficiency to optimal LPSI (default: TRUE)
GAY	Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation:

Index coefficients: $b = w$

The Base Index is appropriate when: - Covariance estimates are unreliable - Computational simplicity is required - A baseline for comparison is needed

Value

List with:

- summary - Data frame with coefficients and metrics
- b - Vector of Base Index coefficients (equal to w)
- w - Named vector of economic weights
- Delta_G - Named vector of expected genetic gains per trait
- lpsi_comparison - Optional comparison with Smith-Hazel LPSI

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
weights <- c(10, 8, 6, 4, 2, 1, 1)

result <- base_index(pmat, gmat, weights, compare_to_lpsi = TRUE)
print(result)

## End(Not run)
```

clgsi *Combined Linear Genomic Selection Index (CLGSI)*

Description

Implements the Combined Linear Genomic Selection Index where selection combines both phenotypic observations and Genomic Estimated Breeding Values (GEBVs). This is used for selecting candidates with both phenotype and genotype data (e.g., in a training population).

Usage

```
clgsi(
  phen_mat = NULL,
  gebv_mat = NULL,
  pmat,
  gmat,
  wmat,
  wcol = 1,
  P_y = NULL,
  P_g = NULL,
  P_yg = NULL,
  reliability = NULL,
  selection_intensity = 2.063,
  GAY = NULL
)
```

Arguments

phen_mat	Matrix of phenotypes (n_genotypes x n_traits). Can be NULL if P_y and P_yg are provided.
gebv_mat	Matrix of GEBVs (n_genotypes x n_traits). Can be NULL if P_g and P_yg are provided.
pmat	Phenotypic variance-covariance matrix (n_traits x n_traits)
gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
wmat	Economic weights matrix (n_traits x k), or vector
wcol	Weight column to use if wmat has multiple columns (default: 1)
P_y	Optional. Phenotypic variance-covariance matrix computed from data (n_traits x n_traits). If NULL (default), uses pmat or computes from phen_mat using cov().
P_g	Optional. GEBV variance-covariance matrix (n_traits x n_traits). If NULL (default), computed from gebv_mat using cov().
P_yg	Optional. Covariance matrix between phenotypes and GEBVs (n_traits x n_traits). If NULL (default), computed from phen_mat and gebv_mat using cov().
reliability	Optional. Reliability of GEBVs (r^2, the squared correlation). See lgsi() for details.

`selection_intensity`
: Selection intensity i (default: 2.063 for 10% selection)

`GAY`
: Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation:

The CLGSI combines phenotypic and genomic information:

$$I = \mathbf{b}_y'\mathbf{y} + \mathbf{b}_g'\hat{\mathbf{g}}$$

Coefficients are obtained by solving the partitioned system:

$$\begin{bmatrix} \mathbf{b}_y \\ \mathbf{b}_g \end{bmatrix} = \begin{bmatrix} \mathbf{P} & \mathbf{P}_{y\hat{g}} \\ \mathbf{P}'_{y\hat{g}} & \mathbf{P}_{\hat{g}} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{G} \\ \mathbf{C}_{\hat{g}g} \end{bmatrix} \mathbf{w}$$

Where: - $\mathbf{P}$ = Var(phenotypes) - $\mathbf{P}_{\hat{g}}$ = Var(GEBVs) - $\mathbf{P}_{y\hat{g}}$ = Cov(phenotypes, GEBVs) - $\mathbf{G}$ = Genotypic variance-covariance - $\mathbf{C}_{\hat{g}g}$ = Cov(GEBV, true BV)

Value

List with components:

- `b_y` - Coefficients for phenotypes
- `b_g` - Coefficients for GEBVs
- `b_combined` - Full coefficient vector [b_y; b_g]
- `P_combined` - Combined variance matrix
- `Delta_H` - Expected genetic advance per trait
- `GA` - Overall genetic advance
- `PRE` - Percent relative efficiency
- `hI2` - Index heritability
- `rHI` - Index accuracy
- `summary` - Data frame with all metrics

References

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing.

Examples

```
## Not run:
# Generate example data
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate phenotypes and GEBVs
```

```
set.seed(123)
n_genotypes <- 100
n_traits <- ncol(gmat)

phen_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 15, sd = 3),
  nrow = n_genotypes, ncol = n_traits
)
gebv_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 10, sd = 2),
  nrow = n_genotypes, ncol = n_traits
)
colnames(phen_mat) <- colnames(gmat)
colnames(gebv_mat) <- colnames(gmat)

# Economic weights
weights <- c(10, 5, 3, 3, 5, 8, 4)

# Calculate CLGSI
result <- clgsi(phen_mat, gebv_mat, pmat, gmat, weights, reliability = 0.7)
print(result$summary)

## End(Not run)
```

---

cppg_lgsi *Combined Predetermined Proportional Gains Linear Genomic Selection Index (CPPG-LGSI)*

Description

Implements the CPPG-LGSI which combines phenotypic and genomic information while achieving predetermined proportional gains between traits. This is the most general constrained genomic index.

Usage

```
cppg_lgsi(
  T_C = NULL,
  Psi_C = NULL,
  d,
  phen_mat = NULL,
  gebv_mat = NULL,
  pmat = NULL,
  gmat = NULL,
  wmat = NULL,
  wcol = 1,
  U = NULL,
  reliability = NULL,
  k_I = 2.063,
  L_I = 1,
  GAY = NULL
)
```

Arguments

T_C	Combined variance-covariance matrix (2t x 2t)
Psi_C	Combined genetic covariance matrix (2t x t)
d	Vector of desired proportional gains (length n_traits or n_constraints)
phen_mat	Optional. Matrix of phenotypes for automatic T_C computation
gebv_mat	Optional. Matrix of GEBVs for automatic T_C computation
pmat	Optional. Phenotypic variance-covariance matrix
gmat	Optional. Genotypic variance-covariance matrix
wmat	Optional. Economic weights for GA/PRE calculation
wcol	Weight column to use if wmat has multiple columns (default: 1)
U	Optional. Constraint matrix (n_traits x n_constraints)
reliability	Optional. Reliability of GEBVs (r^2)
k_I	Selection intensity (default: 2.063)
L_I	Standardization constant (default: 1)
GAY	Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation (Chapter 6, Section 6.4):

Coefficient vector: $beta_C P = beta_C R + theta_C P * delta_C P$

Where beta_CR is from CRLGSI and:

$$theta_C P = (beta'_C * Phi_C * (Phi'_C * T_C^{-1} * Phi_C)^{-1} * d_C)/(d'_C * (Phi'_C * T_C^{-1} * Phi_C)^{-1} * d_C)$$

Selection response: $R_C P = (k_I/L_I) * sqrt(beta_C P' * T_C * beta_C P)$

Value

List with:

- summary - Data frame with coefficients and metrics
- b - Vector of CPPG-LGSI coefficients (β_{CP})
- b_y - Coefficients for phenotypes
- b_g - Coefficients for GEBVs
- E - Expected genetic gains per trait
- theta_CP - Proportionality constant
- gain_ratios - Ratios of achieved to desired gains

Examples

```
## Not run:
# Simulate data
set.seed(123)
n_genotypes <- 100
n_traits <- 5

phen_mat <- matrix(rnorm(n_genotypes * n_traits, 15, 3), n_genotypes, n_traits)
gebv_mat <- matrix(rnorm(n_genotypes * n_traits, 10, 2), n_genotypes, n_traits)

gmat <- cov(phen_mat) * 0.6
pmat <- cov(phen_mat)

# Desired proportional gains
d <- c(2, 1, 1, 0.5, 0)

w <- c(10, 8, 6, 4, 2)

result <- cppg_lgsi(
  phen_mat = phen_mat, gebv_mat = gebv_mat,
  pmat = pmat, gmat = gmat, d = d, wmat = w,
  reliability = 0.7
)
print(result$summary)
print(result$gain_ratios)

## End(Not run)
```

---

cpp_math_primitives *Generic C++ Math Primitives for Experimental Design Statistics*

Description

Generic mathematical operations optimized with C++/Eigen. No design-specific logic - purely mathematical primitives that can be orchestrated by R code to implement any experimental design.

This architecture allows: - Easy addition of new experimental designs (in R only) - C++ speed for heavy computation - Single source of truth (design_stats.R) - Better maintainability and testability

---

crlgsi *Combined Restricted Linear Genomic Selection Index (CRLGSI)*

Description

Implements the CRLGSI which combines phenotypic and genomic information with restrictions on genetic gains. This extends CLGSI to include constraints.

Usage

```
crlgsi(
  T_C = NULL,
  Psi_C = NULL,
  phen_mat = NULL,
  gebv_mat = NULL,
  pmat = NULL,
  gmat = NULL,
  wmat,
  wcol = 1,
  restricted_traits = NULL,
  U = NULL,
  reliability = NULL,
  k_I = 2.063,
  L_I = 1,
  GAY = NULL
)
```

Arguments

T_C	Combined variance-covariance matrix (2t x 2t) where t = n_traits. Structure: [P, P_yg; P_yg', P_g] where P = phenotypic var, P_g = GEBV var, P_yg = covariance between phenotypes and GEBVs. Can be computed automatically if phen_mat and gebv_mat are provided.
Psi_C	Combined genetic covariance matrix (2t x t). Structure: [G; C_gebv_g] where G = genetic var, C_gebv_g = Cov(GEBV, g). Can be computed automatically if gmat and reliability are provided.
phen_mat	Optional. Matrix of phenotypes (n_genotypes x n_traits)
gebv_mat	Optional. Matrix of GEBVs (n_genotypes x n_traits)
pmat	Optional. Phenotypic variance-covariance matrix
gmat	Optional. Genotypic variance-covariance matrix
wmat	Economic weights matrix (n_traits x k), or vector
wcol	Weight column to use if wmat has multiple columns (default: 1)
restricted_traits	Vector of trait indices to restrict (default: NULL)
U	Constraint matrix (2t x n_constraints for combined traits). Alternative to restricted_traits. Ignored if restricted_traits is provided.
reliability	Optional. Reliability of GEBVs (r^2)
k_I	Selection intensity (default: 2.063)
L_I	Standardization constant (default: 1)
GAY	Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation (Chapter 6, Section 6.3):

The CRLGSI combines phenotypic and genomic data with restrictions.

Coefficient vector: $beta_C R = K_C * beta_C$

Where K_C incorporates the restriction matrix.

Selection response: $R_C R = (k_I / L_I) * sqrt(beta_C R' * T_C * beta_C R)$

Expected gains: $E_C R = (k_I / L_I) * (Psi_C * beta_C R) / sqrt(beta_C R' * T_C * beta_C R)$

Value

List with:

- `summary` - Data frame with coefficients and metrics
- `b` - Vector of CRLGSI coefficients (β_{CR})
- `b_y` - Coefficients for phenotypes
- `b_g` - Coefficients for GEBVs
- `E` - Expected genetic gains per trait
- `R` - Overall selection response

Examples

```
## Not run:
# Simulate data
set.seed(123)
n_genotypes <- 100
n_traits <- 5

phen_mat <- matrix(rnorm(n_genotypes * n_traits, 15, 3), n_genotypes, n_traits)
gebv_mat <- matrix(rnorm(n_genotypes * n_traits, 10, 2), n_genotypes, n_traits)

gmat <- cov(phen_mat) * 0.6 # Genotypic component
pmat <- cov(phen_mat)

w <- c(10, 8, 6, 4, 2)

# Restrict traits 2 and 4
result <- crlgsi(
  phen_mat = phen_mat, gebv_mat = gebv_mat,
  pmat = pmat, gmat = gmat, wmat = w,
  restricted_traits = c(2, 4), reliability = 0.7
)
print(result$summary)

## End(Not run)
```

dg_lpsi *Desired Gains Index (DG-LPSI)*

Description

Implements the Pesek & Baker (1969) Desired Gains Index where breeders specify target genetic gains instead of economic weights. This enhanced version includes calculation of implied economic weights and feasibility checking.

Usage

```
dg_lpsi(
  pmat,
  gmat,
  d,
  return_implied_weights = TRUE,
  check_feasibility = TRUE,
  selection_intensity = 2.063
)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits)
gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
d	Vector of desired genetic gains (length n_traits). Example: d = c(1.5, 0.8, -0.2) means gain +1.5 in trait 1, +0.8 in trait 2, -0.2 in trait 3.
return_implied_weights	Logical - calculate implied economic weights? (default: TRUE)
check_feasibility	Logical - warn if desired gains are unrealistic? (default: TRUE)
selection_intensity	Selection intensity i (default: 2.063)

Details

Mathematical Formulation:

1. Index coefficients: $\mathbf{b} = \mathbf{G}^{-1}\mathbf{d}$

2. Expected response: $\Delta\mathbf{G} = (i/\sigma_I)\mathbf{G}\mathbf{b}$

CRITICAL: Scale Invariance Property

The achieved gains $\Delta\mathbf{G}$ are determined by selection intensity (i), genetic variance (G), and phenotypic variance (P), NOT by scaling $\mathbf{b}$. If you multiply $\mathbf{b}$ by constant c, σ_I also scales by c, causing complete cancellation in $\Delta\mathbf{G} = (i/(c\sigma_I))\mathbf{G}(c\mathbf{b}) = (i/\sigma_I)\mathbf{G}\mathbf{b}$.

What DG-LPSI Actually Achieves:

- Proportional gains matching the RATIOS in d (not absolute magnitudes) - Achieved magnitude depends on biological/genetic constraints - Use feasibility checking to verify if desired gains are realistic

3. Implied economic weights (Section 1.4 of Chapter 4):

$$\hat{\mathbf{w}} = \mathbf{G}^{-1}\mathbf{P}\mathbf{b}$$

The implied weights represent the economic values that would have been needed in a Smith-Hazel index to achieve the desired gain PROPORTIONS. Large implied weights indicate traits that are "expensive" to improve (low heritability or unfavorable correlations), while small weights indicate traits that are "cheap" to improve.

Feasibility Checking:

The function estimates maximum possible gains as approximately 3.0 * sqrt(G_ii) (assuming very intense selection with i ~ 3.0) and warns if desired gains exceed 80% of these theoretical maxima.

Value

List with:

- `summary` - Data frame with coefficients, metrics, and implied weights
- `b` - Vector of selection index coefficients
- `Delta_G` - Named vector of achieved genetic gains per trait
- `desired_gains` - Named vector of desired gains (input d)
- `gain_errors` - Difference between desired and achieved gains
- `implied_weights` - Economic weights that would achieve these gains in Smith-Hazel LPSI
- `implied_weights_normalized` - Normalized implied weights (max absolute = 1)
- `feasibility` - Data frame with feasibility analysis per trait
- `hI2` - Index heritability
- `rHI` - Index accuracy

References

Pesek, J., & Baker, R. J. (1969). Desired improvement in relation to selection indices. *Canadian Journal of Plant Science*, 49(6), 803-804.

Examples

```
# Load data
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Specify desired gains (e.g., increase each trait by 1 unit)
desired_gains <- rep(1, ncol(gmat))

# Calculate Desired Gains Index with all enhancements
result <- dg_lpsi(pmat, gmat, d = desired_gains)
```

```
# View summary
print(result$summary)

# Extract implied weights to understand relative "cost" of gains
print(result$implied_weights_normalized)

# Check feasibility
print(result$feasibility)
```

`esim` *Linear Phenotypic Eigen Selection Index (ESIM)*

Description

Implements the ESIM by maximising the squared accuracy ρ^2_{HI} through the generalised eigenproblem of the multi-trait heritability matrix $\mathbf{P}^{-1}\mathbf{C}$.

Unlike the Smith-Hazel LPSI, **no economic weights are required**. The net genetic merit vector $\mathbf{w}_E$ is instead implied by the solution.

Usage

```
esim(pmat, gmat, selection_intensity = 2.063, n_indices = 1L)
```

Arguments

`pmat`	Phenotypic variance-covariance matrix (n_traits x n_traits).
`gmat`	Genotypic variance-covariance matrix (n_traits x n_traits). Corresponds to $\mathbf{C}$ in the Chapter 7 notation.
`selection_intensity`	Selection intensity constant k_I (default: 2.063 for 10% selection).
`n_indices`	Number of leading ESIM vectors to return (default: 1). Returning >1 provides a ranked set of indices for comparative analysis.

Details

Eigenproblem (Section 7.1):

$$(\mathbf{P}^{-1}\mathbf{C} - \lambda^2_E\mathbf{I})\mathbf{b}_E = 0$$

The solution λ^2_E (largest eigenvalue) equals the maximum achievable index heritability $h^2_{I_E}$.

Key metrics:

$$R_E = k_I\sqrt{\mathbf{b}'_E\mathbf{P}\mathbf{b}_E}$$

$$\mathbf{E}_E = k_I\frac{\mathbf{C}\mathbf{b}_E}{\sqrt{\mathbf{b}'_E\mathbf{P}\mathbf{b}_E}}$$

Implied economic weights:

$$\mathbf{w}_E = \frac{\sqrt{\lambda^2_E}}{\mathbf{b}'_E\mathbf{P}\mathbf{b}_E}\mathbf{C}^{-1}\mathbf{P}\mathbf{b}_E$$

Uses cpp_symmetric_solve and cpp_quadratic_form_sym from math_primitives.cpp for efficient matrix operations, and R's eigen() for the eigendecomposition.

Value

Object of class "esim", a list with:

summary Data frame with b coefficients, hI2, rHI, sigma_I, Delta_G, and lambda2 for each index requested.

b Named numeric vector of optimal ESIM coefficients (1st index).

Delta_G Named numeric vector of expected genetic gains per trait.

sigma_I Standard deviation of the index σ_I.

hI2 Index heritability $h^2_{I_E}$ (= leading eigenvalue).

rHI Accuracy r_{HI_E}.

lambda2 Leading eigenvalue (maximised index heritability).

implied_w Implied economic weights $\mathbf{w}_E$.

all_eigenvalues All eigenvalues of $\mathbf{P}^{-1}\mathbf{C}$.

selection_intensity Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 7.1.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

result <- esim(pmat, gmat)
print(result)
summary(result)

## End(Not run)
```

---

estimate_missing_values

Estimate Missing Values in Experimental Data

Description

Estimates and imputes missing values in randomized complete block design (RCBD), Latin square design (LSD), or split plot design (SPD) experimental data.

Uses one of six methods: REML, Yates, Healy, Regression, Mean, or Bartlett.

Usage

```
estimate_missing_values(
  data,
  genotypes,
  replications,
  columns = NULL,
  main_plots = NULL,
  design = c("RCBD", "LSD", "SPD"),
  method = c("REML", "Yates", "Healy", "Regression", "Mean", "Bartlett"),
  tolerance = 1e-06
)
```

Arguments

`data`	Matrix or data.frame with observations (rows) by traits (columns). May contain missing values (NA, NaN, Inf).
`genotypes`	Vector indicating genotype/treatment for each observation (sub-plot treatments in SPD).
`replications`	Vector indicating replication/block (RCBD) or row (LSD) for each observation.
`columns`	Vector indicating column index for each observation (required for Latin Square Design only).
`main_plots`	Vector indicating main plot treatment for each observation (required for Split Plot Design only).
`design`	Character string specifying experimental design: • "RCBD" - Randomized Complete Block Design (default) • "LSD" - Latin Square Design • "SPD" - Split Plot Design
`method`	Character string specifying the estimation method: • **REML** - Restricted Maximum Likelihood with BLUP. Most robust for complex missing patterns. (RCBD, LSD only) • **Yates** - Traditional iterative formula. Simple and fast. Good for simple missing patterns. (RCBD, LSD only) • **Healy** - Healy & Westmacott weighted adjustment method. More stable than Yates for multiple missing values. (RCBD, LSD only) • **Regression** - Linear regression with QR decomposition. Non-iterative, fast and stable. (RCBD, LSD only) • **Mean** - Mean substitution using treatment and block effects. Non-iterative, fastest. (RCBD, LSD, SPD) • **Bartlett** - ANCOVA using other traits as covariates. Best when traits are correlated. (RCBD, LSD only)
`tolerance`	Numeric convergence criterion for iterative methods. Iteration stops when maximum change in estimated values falls below this threshold. Default: 1e-6.

Details

The function handles missing values by iteratively estimating them based on the experimental design structure:

RCBD: 2-way blocking (genotypes × blocks) **LSD:** 3-way blocking (genotypes × rows × columns) **SPD:** Nested structure (blocks > main plots > sub-plots)

Method Availability:

- RCBD: All methods (REML, Yates, Healy, Regression, Mean, Bartlett)
- LSD: All methods (REML, Yates, Healy, Regression, Mean, Bartlett)
- SPD: Mean only (other methods fall back to Mean)

Method Selection Guide:

- Use **REML** for complex missing patterns or when precision is critical
- Use **Yates** for balanced designs with few missing values
- Use **Healy** when multiple values missing from same treatment/block
- Use **Regression** for fast, deterministic estimation
- Use **Mean** for quick estimation when precision is less critical
- Use **Bartlett** when traits are highly correlated

The function uses the centralized design_stats engine for all ANOVA computations, ensuring consistency with gen_varcov(), phen_varcov(), and mean_performance().

Value

Matrix of the same dimensions as `data` with all missing values replaced by estimates.

References

Yates, F. (1933). The analysis of replicated experiments when the field results are incomplete. *Empire Journal of Experimental Agriculture*, 1, 129-142.

Healy, M. J. R., & Westmacott, M. (1956). Missing values in experiments analysed on automatic computers. *Applied Statistics*, 5(3), 203-206.

Bartlett, M. S. (1937). Some examples of statistical methods of research in agriculture and applied biology. *Supplement to the Journal of the Royal Statistical Society*, 4(2), 137-183.

Examples

```
# RCBD example with missing values
data(seldata)
test_data <- seldata[, 3:5]
test_data[c(1, 10, 25), 1] <- NA
test_data[c(5, 15), 2] <- NA

# Impute using Yates method
imputed <- estimate_missing_values(test_data, seldata$treat, seldata$rep, method = "Yates")
```

```
# Check that no NA remain
anyNA(imputed) # Should be FALSE

## Not run:
# Latin Square Design example
# lsd_data should have genotypes, rows, and columns
imputed_lsd <- estimate_missing_values(
  data = lsd_data[, 3:7],
  genotypes = lsd_data$treat,
  replications = lsd_data$row,
  columns = lsd_data$col,
  design = "LSD",
  method = "REML"
)

# Split Plot Design example
# spd_data should have sub-plots, blocks, and main plots
imputed_spd <- estimate_missing_values(
  data = spd_data[, 3:7],
  genotypes = spd_data$subplot,
  replications = spd_data$block,
  main_plots = spd_data$mainplot,
  design = "SPD",
  method = "Mean"
)

## End(Not run)
```

`genetic_genomic_varcov` *Genetic-Genomic Variance-Covariance Matrix (A)*

Description

Computes the genetic-genomic covariance matrix (A) as defined in Chapter 8 (Equation 8.12) for GESIM and related genomic eigen selection indices.

Structure: A = [[C, C_g-gamma], [C_gamma-g, Γ]] (2t x 2t, square symmetric)

where: - C = Var(g) = true genotypic variance-covariance (t x t) - Γ = Var(γ) = genomic variance-covariance (t x t) - C_g-gamma = Cov(g, γ) = covariance between true BVs and GEBVs (t x t) - C_gamma-g = Cov(γ, g) = transpose of C_g-gamma (t x t)

Usage

```
genetic_genomic_varcov(
  gmat,
  Gamma = NULL,
  reliability = NULL,
  C_gebv_g = NULL,
  square = TRUE
)
```

Arguments

gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
Gamma	Genomic variance-covariance matrix (n_traits x n_traits). If NULL, assumed equal to gmat (perfect prediction).
reliability	Optional. Reliability of GEBVs (r^2 = squared correlation between GEBV and true BV). Can be: - Single value (applied to all traits) - Vector of length n_traits (one per trait) - NULL (default): assumes C_g,γ = Gamma (unbiased GEBVs with reliability = 1)
C_gebv_g	Optional. Direct specification of Cov(γ, g) matrix (t x t). If provided, overrides reliability parameter.
square	Logical. If TRUE (default), returns (2t × 2t) square matrix as required for GESIM. If FALSE, returns (2t × t) rectangular form for LMSI.

Details

The genetic-genomic matrix relates selection on phenotypes + GEBVs to expected genetic gains.

For GESIM (Chapter 8): Requires the full (2t × 2t) square matrix for the eigenproblem: (Φ^(-1) A - λI)b = 0

For LMSI/CLGSI (Chapter 4): Can use the rectangular (2t × t) form in the equation: b = P^(-1) G w, where G is (2t × t).

When reliability is provided: - $C_{\gamma g}$ = diag($\sqrt{r^2}$)

When reliability is NULL: - $C_{\gamma g}$ = Gamma (assumes unbiased GEBVs, perfect prediction)

Value

Genetic-genomic covariance matrix: - If square = TRUE: (2t × 2t) symmetric matrix for GESIM/eigen indices - If square = FALSE: (2t × t) rectangular matrix for LMSI where t is the number of traits

References

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapters 4 & 8.

Examples

```
## Not run:
# Generate example data
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate genomic covariance
Gamma <- gmat * 0.8

# For GESIM: Get square (2t × 2t) matrix
A_square <- genetic_genomic_varcov(gmat, Gamma, reliability = 0.7)
print(dim(A_square)) # Should be 14 x 14 (2t × 2t)

# For LMSI: Get rectangular (2t × t) matrix
```

```
A_rect <- genetic_genomic_varcov(gmat, Gamma, reliability = 0.7, square = FALSE)
print(dim(A_rect)) # Should be 14 x 7 (2t × t)

## End(Not run)
```

`genomic_varcov` *Genomic Variance-Covariance Matrix (Γ)*

Description

Computes genomic variance-covariance matrix (Γ or Gamma) from a matrix of Genomic Estimated Breeding Values (GEBVs).

γ (gamma) represents GEBV vectors obtained from genomic prediction models (e.g., GBLUP, rrBLUP, Genomic BLUP). This function computes Var(γ) = Γ.

Usage

```
genomic_varcov(gebv_mat, method = "pearson", use = "complete.obs")
```

Arguments

`gebv_mat`	Matrix of GEBVs (n_genotypes x n_traits)
`method`	Character string specifying correlation method: "pearson" (default), "kendall", or "spearman"
`use`	Character string specifying how to handle missing values: "everything" (default), "complete.obs", "pairwise.complete.obs", etc. See `cov` for details.

Details

The genomic variance-covariance matrix Γ captures genetic variation as predicted by molecular markers. It is computed as:

where γ_i is the GEBV vector for genotype i and μ_γ is the mean GEBV vector.

Missing Value Handling: - "complete.obs": Uses only complete observations (recommended) - "pairwise.complete.obs": Uses pairwise-complete observations (may not be PSD) - "everything": Fails if any NA present

When using pairwise deletion, the resulting matrix may not be positive semi-definite (PSD), which can cause numerical issues in selection indices.

Applications: In selection index theory: - Used in LGSI (Linear Genomic Selection Index) - Component of Φ (phenomic-genomic covariance) - Component of A (genetic-genomic covariance)

Value

Symmetric genomic variance-covariance matrix (n_traits x n_traits)

References

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapters 4 & 8.

Examples

```
## Not run:
# Simulate GEBVs
set.seed(123)
n_genotypes <- 100
n_traits <- 5
gebv_mat <- matrix(rnorm(n_genotypes * n_traits),
  nrow = n_genotypes, ncol = n_traits
)
colnames(gebv_mat) <- paste0("Trait", 1:n_traits)

# Compute genomic variance-covariance
Gamma <- genomic_varcov(gebv_mat)
print(Gamma)

## End(Not run)
```

---

gen_advance *Genetic Advance for PRE*

Description

Genetic Advance for PRE

Usage

```
gen_advance(phen_mat, gen_mat, weight_mat)
```

Arguments

phen_mat	phenotypic matrix value of desired characters
gen_mat	genotypic matrix value of desired characters
weight_mat	weight matrix value of desired characters

Value

Genetic advance of character or character combinations

Examples

```
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
gen_advance(phen_mat = pmat[1, 1], gen_mat = gmat[1, 1], weight_mat = weight[1, 2])
```

gen_varcov *Genotypic Variance-Covariance Analysis*

Description

Genotypic Variance-Covariance Analysis

Usage

```
gen_varcov(
  data,
  genotypes,
  replication,
  columns = NULL,
  main_plots = NULL,
  design_type = c("RCBD", "LSD", "SPD"),
  method = c("REML", "Yates", "Healy", "Regression", "Mean", "Bartlett")
)
```

Arguments

data	traits to be analyzed
genotypes	vector containing genotypes/treatments (sub-plot treatments in SPD)
replication	vector containing replication/blocks (RCBD) or rows (LSD)
columns	vector containing columns (required for Latin Square Design only)
main_plots	vector containing main plot treatments (required for Split Plot Design only)
design_type	experimental design type: "RCBD" (default), "LSD" (Latin Square), or "SPD" (Split Plot)
method	Method for missing value imputation: "REML" (default), "Yates", "Healy", "Regression", "Mean", or "Bartlett"

Value

A Genotypic Variance-Covariance Matrix

Examples

```
# RCBD example
gen_varcov(data = seldata[, 3:9], genotypes = seldata$treat, replication = seldata$rep)

# Latin Square Design example (requires columns parameter)
# gen_varcov(data=lsd_data[,3:7], genotypes=lsd_data$treat,
#           replication=lsd_data$row, columns=lsd_data$col, design_type="LSD")

# Split Plot Design example (requires main_plots parameter)
# gen_varcov(data=spd_data[,3:7], genotypes=spd_data$subplot,
#           replication=spd_data$block, main_plots=spd_data$mainplot, design_type="SPD")
```

gesim *Linear Genomic Eigen Selection Index Method (GESIM)*

Description

Implements the GESIM by maximising the squared accuracy through the generalised eigenproblem combining phenotypic data with GEBVs (Genomic Estimated Breeding Values). No economic weights are required.

Usage

```
gesim(pmat, gmat, Gamma, selection_intensity = 2.063)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits).
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).
Gamma	Covariance between phenotypes and GEBVs (n_traits x n_traits). This represents Cov(y, gamma) where gamma are GEBVs.
selection_intensity	Selection intensity constant k_I (default: 2.063 for 10% selection).

Details

Eigenproblem (Section 8.2):

$$(\mathbf{\Phi}^{-1}\mathbf{A} - \lambda_G^2 \mathbf{I}_{2t})\boldsymbol{\beta}_G = 0$$

where:

$$\mathbf{\Phi} = \begin{bmatrix} \mathbf{P} & \mathbf{\Gamma} \\ \mathbf{\Gamma} & \mathbf{\Gamma} \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{C} & \mathbf{\Gamma} \\ \mathbf{\Gamma} & \mathbf{\Gamma} \end{bmatrix}$$

Implied economic weights:

$$\mathbf{w}_G = \mathbf{A}^{-1}\mathbf{\Phi}\boldsymbol{\beta}$$

Selection response:

$$R_G = k_I \sqrt{\boldsymbol{\beta}_G' \mathbf{\Phi} \boldsymbol{\beta}_G}$$

Expected genetic gain per trait:

$$\mathbf{E}_G = k_I \frac{\mathbf{A}\boldsymbol{\beta}_G}{\sqrt{\boldsymbol{\beta}_G' \mathbf{\Phi} \boldsymbol{\beta}_G}}$$

Value

Object of class "gesim", a list with:

summary Data frame with coefficients and metrics.

b_y Coefficients for phenotypic data.

b_gamma Coefficients for GEBVs.

b_combined Combined coefficient vector.

E_G Expected genetic gains per trait.

sigma_I Standard deviation of the index.

hI2 Index heritability (= leading eigenvalue).

rHI Accuracy r_{HI}.

R_G Selection response.

lambda2 Leading eigenvalue.

implied_w Implied economic weights.

selection_intensity Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 8.2.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate GEBV covariance (in practice, compute from genomic predictions)
Gamma <- gmat * 0.8 # Assume 80% GEBV-phenotype covariance

result <- gesim(pmat, gmat, Gamma)
print(result)

## End(Not run)
```

---

gw_esim *Genome-Wide Linear Eigen Selection Index Method (GW-ESIM)*

Description

Implements the GW-ESIM by incorporating genome-wide marker effects directly into the eigen selection index framework. Uses N marker scores alongside phenotypic data.

Usage

```
gw_esim(pmat, gmat, G_M, M, selection_intensity = 2.063)
```

Arguments

`pmat`	Phenotypic variance-covariance matrix (n_traits x n_traits).
`gmat`	Genotypic variance-covariance matrix (n_traits x n_traits).
`G_M`	Covariance between phenotypes and marker scores (n_traits x N_markers).
`M`	Variance-covariance matrix of marker scores (N_markers x N_markers).
`selection_intensity`	Selection intensity constant k_I (default: 2.063 for 10% selection).

Details

Eigenproblem (Section 8.3):

$$(\mathbf{Q}^{-1}\mathbf{X} - \lambda_W^2 \mathbf{I}_{(t+N)})\boldsymbol{\beta}_W = 0$$

where:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{P} & \mathbf{G}_M \\ \mathbf{G}_M' & \mathbf{M} \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{C} & \mathbf{G}_M \\ \mathbf{G}_M' & \mathbf{M} \end{bmatrix}$$

Selection response:

$$R_W = k_I \sqrt{\boldsymbol{\beta}_W' \mathbf{Q} \boldsymbol{\beta}_W}$$

Expected genetic gain per trait:

$$\mathbf{E}_W = k_I \frac{\mathbf{X}\boldsymbol{\beta}_W}{\sqrt{\boldsymbol{\beta}_W' \mathbf{Q} \boldsymbol{\beta}_W}}$$

Value

Object of class `"gw_esim"`, a list with:

`summary` Data frame with key metrics.

`b_y` Coefficients for phenotypic data.

`b_m` Coefficients for marker scores.

`b_combined` Combined coefficient vector.

`E_W` Expected genetic gains per trait.

`sigma_I` Standard deviation of the index.

`hI2` Index heritability (= leading eigenvalue).

`rHI` Accuracy.

`R_W` Selection response.

`lambda2` Leading eigenvalue.

`selection_intensity` Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 8.3.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate marker data
N_markers <- 100
n_traits <- nrow(gmat)
G_M <- matrix(rnorm(n_traits * N_markers, sd = 0.5), n_traits, N_markers)
M <- diag(N_markers) + matrix(rnorm(N_markers^2, sd = 0.1), N_markers, N_markers)
M <- (M + t(M)) / 2 # Make symmetric

result <- gw_esim(pmat, gmat, G_M, M)
print(result)

## End(Not run)
```

---

gw_lmsi *Genome-Wide Linear Marker Selection Index (GW-LMSI)*

Description

Implements the GW-LMSI which uses all available genome-wide markers directly as predictors in the selection index. Unlike LMSI which uses aggregated marker scores per trait, GW-LMSI treats each individual marker as a separate predictor.

Usage

```
gw_lmsi(
  marker_mat,
  trait_mat = NULL,
  gmat,
  P_GW = NULL,
  G_GW = NULL,
  wmat,
  wcol = 1,
  lambda = 0,
  selection_intensity = 2.063,
  GAY = NULL
)
```

Arguments

marker_mat	Matrix of marker genotypes (n_genotypes x n_markers). Typically coded as -1, 0, 1 or 0, 1, 2.
trait_mat	Matrix of trait values (n_genotypes x n_traits). Used to compute G_GW if not provided.
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).
P_GW	Marker covariance matrix (n_markers x n_markers). If NULL, computed as Var(marker_mat).
G_GW	Covariance between markers and traits (n_markers x n_traits). If NULL, computed as Cov(marker_mat, trait_mat).
wmat	Economic weights matrix (n_traits x k), or vector.
wcol	Weight column to use if wmat has multiple columns (default: 1).
lambda	Ridge regularization parameter (default: 0). If lambda > 0, uses P_GW + lambda*I for regularization. Automatic warnings issued when n_markers > n_genotypes (high-dimensional case) or when P_GW is ill-conditioned. Recommended values: 0.01-0.1 times mean(diag(P_GW)).
selection_intensity	Selection intensity k (default: 2.063 for 10% selection).
GAY	Optional. Genetic advance of comparative trait for PRE calculation.

Details

Mathematical Formulation:

The GW-LMSI maximizes the correlation between the index $I_{GW} = \mathbf{b}'_{GW}\mathbf{m}$ and the breeding objective $H = \mathbf{w}'\mathbf{g}$.

Marker covariance matrix:

$$\mathbf{P}_{GW} = Var(\mathbf{m})$$

Covariance between markers and traits:

$$\mathbf{G}_{GW} = Cov(\mathbf{m}, \mathbf{g})$$

Index coefficients (with optional Ridge regularization):

$$\mathbf{b}_{GW} = (\mathbf{P}_{GW} + \lambda\mathbf{I})^{-1}\mathbf{G}_{GW}\mathbf{w}$$

Accuracy:

$$\rho_{HI} = \sqrt{\frac{\mathbf{b}'_{GW}\mathbf{G}_{GW}\mathbf{w}}{\mathbf{w}'\mathbf{G}\mathbf{w}}}$$

Selection response:

$$R_{GW} = k\sqrt{\mathbf{b}'_{GW}\mathbf{P}_{GW}\mathbf{b}_{GW}}$$

Expected genetic gain per trait:

$$\mathbf{E}_{GW} = k\frac{\mathbf{G}'_{GW}\mathbf{b}_{GW}}{\sigma_{I_{GW}}}$$

Note on Singularity Detection and Regularization: The function automatically detects problematic cases:

1. **High-dimensional case**: When n_markers > n_genotypes, P_GW is mathematically singular (rank-deficient). The function issues a warning and suggests an appropriate lambda value.

2. **Ill-conditioned case**: When P_GW has a high condition number (> 1e10), indicating numerical instability.

3. **Numerical singularity**: When P_GW has eigenvalues near zero.

Ridge regularization adds λI to P_GW, ensuring positive definiteness. Recommended lambda values are 0.01-0.1 times the average diagonal element of P_GW. Users can also set lambda = 0 to force generalized inverse (less stable but sometimes needed).

Value

List of class `"gw_lmsi"` with components:

`b` Index coefficients for markers (n_markers).

`P_GW` Marker covariance matrix (n_markers x n_markers).

`G_GW` Covariance between markers and traits (n_markers x n_traits).

`rHI` Index accuracy (correlation between index and breeding objective).

`sigma_I` Standard deviation of the index.

`R` Selection response (k * sigma_I).

`Delta_H` Expected genetic gain per trait (vector of length n_traits).

`GA` Overall genetic advance in breeding objective.

`PRE` Percent relative efficiency (if GAY provided).

`hI2` Index heritability.

`lambda` Ridge regularization parameter used.

`n_markers` Number of markers.

`high_dimensional` Logical indicating if n_markers > n_genotypes.

`condition_number` Condition number of P_GW (if computed).

`summary` Data frame with metrics.

References

Lande, R., & Thompson, R. (1990). Efficiency of marker-assisted selection in the improvement of quantitative traits. Genetics, 124(3), 743-756.

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapter 4.

Examples

```
## Not run:
# Load data
data(seldata)
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate marker data
set.seed(123)
n_genotypes <- 100
n_markers <- 200
n_traits <- ncol(gmat)

# Marker matrix (coded as 0, 1, 2)
marker_mat <- matrix(sample(0:2, n_genotypes * n_markers, replace = TRUE),
  nrow = n_genotypes, ncol = n_markers
)

# Trait matrix
trait_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 15, sd = 3),
  nrow = n_genotypes, ncol = n_traits
)

# Economic weights
weights <- c(10, 5, 3, 3, 5, 8, 4)

# Calculate GW-LMSI with Ridge regularization
result <- gw_lmsi(marker_mat, trait_mat, gmat,
  wmat = weights, lambda = 0.01
)
print(result$summary)

## End(Not run)
```

---

haldane_mapping *Haldane's Mapping Function*

Description

Converts genetic distance (in Morgans) to recombination fraction using Haldane's mapping function. This function models the relationship between genetic distance and the probability of recombination between loci.

Usage

```
haldane_mapping(distance)
```

Arguments

distance	Genetic distance in Morgans (scalar or vector). One Morgan corresponds to a 50% recombination frequency.

Details

Mathematical Formula (Chapter 10, Section 10.1):

The relationship between recombination fraction (r) and genetic distance (d):

$$r = \frac{1}{2}(1 - e^{-2d})$$

Where: - d = Genetic distance in Morgans - r = Recombination fraction (probability of recombination per meiosis)

This function assumes no crossover interference beyond that implied by the mapping function itself.

Value

Recombination fraction (r) ranging from 0 to 0.5. - r = 0 indicates complete linkage (no recombination) - r = 0.5 indicates independent assortment (unlinked loci)

Examples

```
# Zero distance means complete linkage (no recombination)
haldane_mapping(0) # Returns 0

# 1 Morgan distance
haldane_mapping(1) # Returns ~0.43

# Large distance approaches 0.5 (independent assortment)
haldane_mapping(10) # Returns ~0.5

# Vector of distances
distances <- c(0, 0.1, 0.5, 1.0, 2.0)
haldane_mapping(distances)
```

`inverse_haldane_mapping` *Inverse Haldane Mapping Function*

Description

Converts recombination fraction back to genetic distance (in Morgans). This is the inverse of Haldane's mapping function.

Usage

```
inverse_haldane_mapping(recombination_fraction)
```

Arguments

`recombination_fraction` Recombination fraction (r) between 0 and 0.5.

Details

Mathematical Formula:

Solving Haldane's equation for d:

$$d = -\frac{1}{2}\ln(1 - 2r)$$

Value

Genetic distance in Morgans.

Examples

```
# Convert recombination fraction to distance
inverse_haldane_mapping(0.25) # Returns ~0.347 Morgans
inverse_haldane_mapping(0.5) # Returns Inf (unlinked)
```

---

lgsi *Linear Genomic Selection Index (LGSI)*

Description

Implements the Linear Genomic Selection Index where selection is based solely on Genomic Estimated Breeding Values (GEBVs). This is used for selecting candidates that have been genotyped but not phenotyped (e.g., in a testing population).

Usage

```
lgsi(
  gebv_mat,
  gmat,
  wmat,
  wcol = 1,
  reliability = NULL,
  selection_intensity = 2.063,
  GAY = NULL
)
```

Arguments

gebv_mat	Matrix of GEBVs (n_genotypes x n_traits)
gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
wmat	Economic weights matrix (n_traits x k), or vector
wcol	Weight column to use if wmat has multiple columns (default: 1)
reliability	Optional. Reliability of GEBVs (correlation between GEBV and true BV). Can be: - Single value (applied to all traits) - Vector of length n_traits (one per trait) - NULL (default): estimated from GEBV variance (assumes reliability = GEBV_var / G_var)

`selection_intensity`
: Selection intensity i (default: 2.063 for 10% selection)

`GAY`
: Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation:

The LGSI maximizes the correlation between the index I = b' * gebv and

Index coefficients: $\mathbf{b} = \mathbf{P}_{\hat{g}}^{-1}\mathbf{C}_{\hat{g}g}\mathbf{w}$

Where: - $\mathbf{P}_{\hat{g}}$ = Var(gebv) - variance-covariance of GEBVs - $\mathbf{C}_{\hat{g}g}$ = Cov(gebv, g) - covariance between GEBVs and true breeding values

If reliability (r) is known: $\mathbf{C}_{\hat{g}g} = \text{diag}(r)\mathbf{P}_{\hat{g}}$

Expected response: $\Delta\mathbf{H} = \frac{i}{\sigma_I}\mathbf{C}_{\hat{g}g}\mathbf{b}$

Value

List with components:

- `b` - Index coefficients
- `P_gebv` - GEBV variance-covariance matrix
- `reliability` - Reliability values used
- `Delta_H` - Expected genetic advance per trait
- `GA` - Overall genetic advance in the index
- `PRE` - Percent relative efficiency (if GAY provided)
- `hI2` - Index heritability
- `rHI` - Index accuracy
- `sigma_I` - Standard deviation of the index
- `summary` - Data frame with all metrics

References

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing.

Examples

```
## Not run:
# Generate example data
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate GEBVs (in practice, these come from genomic prediction)
set.seed(123)
n_genotypes <- 100
n_traits <- ncol(gmat)
gebv_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 10, sd = 2),
  nrow = n_genotypes, ncol = n_traits
```

```
)
colnames(gebv_mat) <- colnames(gmat)

# Economic weights
weights <- c(10, 5, 3, 3, 5, 8, 4)

# Calculate LGSI
result <- lgsi(gebv_mat, gmat, weights, reliability = 0.7)
print(result$summary)

## End(Not run)
```

---

lmsi *Linear Marker Selection Index (LMSI)*

Description

Implements the LMSI which combines phenotypic information with molecular marker scores from statistically significant markers (Lande & Thompson, 1990). The index is I = b_y' * y + b_s' * s, where y are phenotypes and s are marker scores.

Usage

```
lmsi(
  phen_mat = NULL,
  marker_scores = NULL,
  pmat,
  gmat,
  G_s = NULL,
  wmat,
  wcol = 1,
  selection_intensity = 2.063,
  GAY = NULL
)
```

Arguments

phen_mat	Matrix of phenotypes (n_genotypes x n_traits). Can be NULL if G_s is provided directly (theoretical case where covariance structure is known without needing empirical data).
marker_scores	Matrix of marker scores (n_genotypes x n_traits). These are computed as s_j = sum(x_jk * beta_jk) where x_jk is the coded marker value and beta_jk is the estimated marker effect for trait j. Can be NULL if G_s is provided directly.
pmat	Phenotypic variance-covariance matrix (n_traits x n_traits).
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).

`G_s` Genomic covariance matrix explained by markers (n_traits x n_traits). This represents Var(s) which approximates Cov(y, s) when markers fully explain genetic variance. If provided, phen_mat and marker_scores become optional as the covariance structure is specified directly. If NULL, computed empirically from marker_scores and phen_mat.

`wmat` Economic weights matrix (n_traits x k), or vector.

`wcol` Weight column to use if wmat has multiple columns (default: 1).

`selection_intensity` Selection intensity k (default: 2.063 for 10% selection).

`GAY` Optional. Genetic advance of comparative trait for PRE calculation.

Details

Mathematical Formulation:

The LMSI maximizes the correlation between the index $I_{LMSI} = \mathbf{b}_y'\mathbf{y} + \mathbf{b}_s'\mathbf{s}$ and the breeding objective $H = \mathbf{w}'\mathbf{g}$.

Combined covariance matrices:

$$\mathbf{P}_L = \begin{bmatrix} \mathbf{P} & \text{Cov}(\mathbf{y}, \mathbf{s}) \\ \text{Cov}(\mathbf{y}, \mathbf{s})' & \text{Var}(\mathbf{s}) \end{bmatrix}$$

$$\mathbf{G}_L = \begin{bmatrix} \mathbf{G} \\ \mathbf{G}_s \end{bmatrix}$$

where $\mathbf{P}$ is the phenotypic variance, $\text{Cov}(\mathbf{y}, \mathbf{s})$ is the covariance between phenotypes and marker scores (computed from data), $\text{Var}(\mathbf{s})$ is the variance of marker scores, $\mathbf{G}$ is the genotypic variance, and $\mathbf{G}_s$ represents the genetic covariance explained by markers.

Index coefficients:

$$\mathbf{b}_{LMSI} = \mathbf{P}_L^{-1}\mathbf{G}_L\mathbf{w}$$

Accuracy:

$$\rho_{HI} = \sqrt{\frac{\mathbf{b}_{LMSI}'\mathbf{G}_L\mathbf{w}}{\mathbf{w}'\mathbf{G}\mathbf{w}}}$$

Selection response:

$$R_{LMSI} = k\sigma_{I_{LMSI}} = k\sqrt{\mathbf{b}_{LMSI}'\mathbf{P}_L\mathbf{b}_{LMSI}}$$

Expected genetic gain per trait:

$$\mathbf{E}_{LMSI} = k\frac{\mathbf{G}_L'\mathbf{b}_{LMSI}}{\sigma_{I_{LMSI}}}$$

Value

List of class `"lmsi"` with components:

`b_y` Coefficients for phenotypes (n_traits vector).

`b_s` Coefficients for marker scores (n_traits vector).

`b_combined` Combined coefficient vector [b_y; b_s] (2*n_traits vector).

`P_L` Combined phenotypic-marker covariance matrix (2*n_traits x 2*n_traits).

`G_L` Combined genetic-marker covariance matrix (2*n_traits x n_traits).

`G_s` Genomic covariance matrix explained by markers (n_traits x n_traits).

`rHI` Index accuracy (correlation between index and breeding objective).

`sigma_I` Standard deviation of the index.

`R` Selection response (k * sigma_I).

`Delta_H` Expected genetic gain per trait (vector of length n_traits).

`GA` Overall genetic advance in breeding objective.

`PRE` Percent relative efficiency (if GAY provided).

`hI2` Index heritability.

`summary` Data frame with coefficients and metrics (combined view).

`phenotype_coeffs` Data frame with phenotype coefficients only.

`marker_coeffs` Data frame with marker score coefficients only.

`coeff_analysis` Data frame with coefficient distribution analysis.

References

Lande, R., & Thompson, R. (1990). Efficiency of marker-assisted selection in the improvement of quantitative traits. Genetics, 124(3), 743-756.

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapter 4.

Examples

```
## Not run:
# Load data
data(seldata)
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate marker scores (in practice, computed from QTL mapping)
set.seed(123)
n_genotypes <- 100
n_traits <- ncol(gmat)
marker_scores <- matrix(rnorm(n_genotypes * n_traits, mean = 5, sd = 1.5),
  nrow = n_genotypes, ncol = n_traits
)
colnames(marker_scores) <- colnames(gmat)

# Simulate phenotypes
phen_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 15, sd = 3),
  nrow = n_genotypes, ncol = n_traits
)
colnames(phen_mat) <- colnames(gmat)

# Economic weights
```

```
weights <- c(10, 5, 3, 3, 5, 8, 4)

# Calculate LMSI
result <- lmsi(phen_mat, marker_scores, pmat, gmat,
  G_s = NULL, wmat = weights
)
print(result$summary)

## End(Not run)
```

---

lpsi *Combinatorial Linear Phenotypic Selection Index*

Description

Build all possible Smith-Hazel selection indices from trait combinations, with optional exclusion of specific traits.

This function systematically evaluates indices for all combinations of ncomb traits, which is useful for identifying the most efficient subset of traits for selection.

Usage

```
lpsi(ncomb, pmat, gmat, wmat, wcol = 1, GAY, excluding_trait = NULL)
```

Arguments

ncomb	Number of traits per combination
pmat	Phenotypic variance-covariance matrix
gmat	Genotypic variance-covariance matrix
wmat	Weight matrix
wcol	Weight column number if more than one weight set (default: 1)
GAY	Genetic advance of comparative trait (optional)
excluding_trait	Optional. Traits to exclude from combinations. Can be: (1) numeric vector of trait indices (e.g., c(1, 3)), (2) character vector of trait names (e.g., c("sypp", "dtf")), (3) data frame/matrix columns with trait data (trait names extracted from column names). When specified, only combinations that do NOT contain any of these traits are returned.

Value

Data frame of all possible selection indices with metrics (GA, PRE, Delta_G, rHI, hI2)

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
wmat <- weight_mat(weight)

# Build all 3-trait indices
result <- lpsi(ncomb = 3, pmat = pmat, gmat = gmat, wmat = wmat, wcol = 1)

# Exclude specific traits
result <- lpsi(
  ncomb = 3, pmat = pmat, gmat = gmat, wmat = wmat,
  excluding_trait = c(1, 3)
)

## End(Not run)
```

---

maize_geno *Synthetic Maize Genomic Data*

Description

A synthetic dataset containing Single Nucleotide Polymorphism (SNP) marker data for the 100 maize genotypes found in maize_pheno. Designed for testing genomic selection indices (GESIM/RGESIM) and relationship matrices.

Usage

```
data(maize_geno)
```

Format

A numeric matrix with 100 rows (genotypes) and 500 columns (SNP markers):

Rows 100 genotypes, matching the Genotype column in maize_pheno.

Columns 500 simulated SNP markers, coded as 0, 1, or 2.

Source

Simulated for the selection.index package to provide reproducible examples.

Examples

```
data(maize_geno)
dim(maize_geno)
```

maize_pheno *Synthetic Maize Phenotypic Data*

Description

A synthetic dataset containing multi-environment phenotypic records for 100 maize genotypes. Designed to demonstrate phenotypic selection index calculations and variance-covariance matrix estimations.

Usage

```
data(maize_pheno)
```

Format

A data frame with 600 rows and 6 variables:

Genotype Factor representing 100 unique maize genotypes.

Environment Factor representing 2 distinct growing environments.

Block Factor representing 3 replicate blocks within each environment.

Yield Numeric vector of grain yield in kg/ha.

PlantHeight Numeric vector of plant height in centimeters.

DaysToMaturity Numeric vector of days to physiological maturity.

Source

Simulated for the selection.index package to provide reproducible examples.

Examples

```
data(maize_pheno)
head(maize_pheno)
```

mean_performance *Mean performance of phenotypic data*

Description

Mean performance of phenotypic data

Usage

```
mean_performance(
  data,
  genotypes,
  replications,
  columns = NULL,
  main_plots = NULL,
  design_type = c("RCBD", "LSD", "SPD"),
  method = c("REML", "Yates", "Healy", "Regression", "Mean", "Bartlett")
)
```

Arguments

data	data for analysis
genotypes	genotypes vector (sub-plot treatments in SPD)
replications	replication vector
columns	vector containing columns (required for Latin Square Design only)
main_plots	vector containing main plot treatments (required for Split Plot Design only)
design_type	experimental design type: "RCBD" (default), "LSD" (Latin Square), or "SPD" (Split Plot)
method	Method for missing value imputation: "REML" (default), "Yates", "Healy", "Regression", "Mean", or "Bartlett"

Value

Dataframe of mean performance analysis

Examples

```
mean_performance(data = seldata[, 3:9], genotypes = seldata[, 2], replications = seldata[, 1])
```

---

mesim *Molecular Eigen Selection Index Method (MESIM)*

Description

Implements the MESIM by maximising the squared accuracy through the generalised eigenproblem combining phenotypic data with molecular marker scores. Unlike Smith-Hazel LPSI, **no economic weights are required**.

Usage

```
mesim(pmat, gmat, S_M, S_Mg = NULL, S_var = NULL, selection_intensity = 2.063)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits).
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).
S_M	Covariance between phenotypes and marker scores (n_traits x n_traits). This represents Cov(y, s) where s are marker scores. Used in the phenotypic variance matrix T_M.
S_Mg	Covariance between genetic values and marker scores (n_traits x n_traits). This represents Cov(g, s). Used in the genetic variance matrix Psi_M. If NULL (default), uses S_M, which is appropriate when assuming Cov(e, s) ~= 0 (errors uncorrelated with markers), so Cov(y,s) ~= Cov(g,s).
S_var	Variance-covariance matrix of marker scores (n_traits x n_traits). Represents Var(s). If NULL (default), uses S_M for backward compatibility, but providing the actual Var(s) is more statistically rigorous.
selection_intensity	Selection intensity constant k_I (default: 2.063 for 10% selection).

Details

Eigenproblem (Section 8.1):

$$(\mathbf{T}_M^{-1}\mathbf{\Psi}_M - \lambda_M^2\mathbf{I}_{2t})\boldsymbol{\beta}_M = 0$$

where:

$$\mathbf{T}_M = \begin{bmatrix} \mathbf{P} & \mathrm{Cov}(\mathbf{y},\mathbf{s}) \\ \mathrm{Cov}(\mathbf{s},\mathbf{y}) & \mathrm{Var}(\mathbf{s}) \end{bmatrix}$$

$$\mathbf{\Psi}_M = \begin{bmatrix} \mathbf{C} & \mathrm{Cov}(\mathbf{g},\mathbf{s}) \\ \mathrm{Cov}(\mathbf{s},\mathbf{g}) & \mathrm{Var}(\mathbf{s}) \end{bmatrix}$$

Theoretical distinction:

- T_M uses phenotypic covariances: Cov(y, s) provided via S_M
- Psi_M uses genetic covariances: Cov(g, s) provided via S_Mg
- Since y = g + e, if Cov(e, s) ~= 0, then Cov(y, s) ~= Cov(g, s)
- Chapter 8.1 assumes marker scores are pure genetic predictors, so S_M can be used for both (default behavior when S_Mg = NULL)

The solution λ_M^2 (largest eigenvalue) equals the maximum achievable index heritability.

Key metrics:

$$R_M = k_I\sqrt{\boldsymbol{\beta}_M'\mathbf{T}_M\boldsymbol{\beta}_M}$$

$$\mathbf{E}_M = k_I\frac{\mathbf{\Psi}_M\boldsymbol{\beta}_M}{\sqrt{\boldsymbol{\beta}_M'\mathbf{T}_M\boldsymbol{\beta}_M}}$$

Value

Object of class `"mesim"`, a list with:

`summary` Data frame with coefficients and metrics.

`b_y` Coefficients for phenotypic data.

`b_s` Coefficients for marker scores.

`b_combined` Combined coefficient vector.

`E_M` Expected genetic gains per trait.

`sigma_I` Standard deviation of the index.

`hI2` Index heritability (= leading eigenvalue).

`rHI` Accuracy r_{HI}.

`R_M` Selection response.

`lambda2` Leading eigenvalue (maximised index heritability).

`selection_intensity` Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 8.1.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate marker score matrices (in practice, compute from data)
S_M <- gmat * 0.7 # Cov(y, s) - phenotype-marker covariance
S_Mg <- gmat * 0.65 # Cov(g, s) - genetic-marker covariance
S_var <- gmat * 0.8 # Var(s) - marker score variance

# Most rigorous: Provide all three covariance matrices
result <- mesim(pmat, gmat, S_M, S_Mg = S_Mg, S_var = S_var)
print(result)

# Standard usage: Cov(g,s) defaults to Cov(y,s) when errors uncorrelated
result_standard <- mesim(pmat, gmat, S_M, S_var = S_var)

# Backward compatible: Chapter 8.1 simplified notation
result_simple <- mesim(pmat, gmat, S_M)

## End(Not run)
```

missing-value *Missing Value Imputation for Experimental Designs*

Description

Exported wrapper for missing value estimation in RCBD, Latin Square, and Split Plot designs. Calls the centralized design_stats engine for ANOVA computations.

mlgsi *Multistage Linear Genomic Selection Index (MLGSI)*

Description

Implements the two-stage Linear Genomic Selection Index where selection is based on GEBVs at both stages with covariance adjustments due to selection effects.

Usage

```
mlgsi(
  Gamma1,
  Gamma,
  A1,
  A,
  C,
  G1,
  P1,
  wmat,
  wcol = 1,
  selection_proportion = 0.1,
  use_young_method = FALSE,
  k1_manual = 2.063,
  k2_manual = 2.063,
  tau = NULL,
  reliability = NULL
)
```

Arguments

Gamma1	GEBV variance-covariance matrix for stage 1 traits (n1 x n1)
Gamma	GEBV variance-covariance matrix for all traits at stage 2 (n x n)
A1	Covariance matrix between GEBVs and true breeding values for stage 1 (n1 x n1)
A	Covariance matrix between GEBVs and true breeding values for stage 2 (n x n1)
C	Genotypic variance-covariance matrix for all traits (n x n)

G1	Genotypic variance-covariance matrix for stage 1 traits (n1 x n1)
P1	Phenotypic variance-covariance matrix for stage 1 traits (n1 x n1)
wmat	Economic weights vector or matrix (n x k)
wcol	Weight column to use if wmat has multiple columns (default: 1)
selection_proportion	Proportion selected at each stage (default: 0.1)
use_young_method	Logical. Use Young's method for selection intensities (default: FALSE). Young's method tends to overestimate intensities; manual intensities are recommended.
k1_manual	Manual selection intensity for stage 1
k2_manual	Manual selection intensity for stage 2
tau	Standardized truncation point
reliability	Optional reliability vector for computing A matrices

Details

Mathematical Formulation:

Stage 1: The genomic index is $I_1 = \beta_1' \gamma_1$

Coefficients: $\beta_1 = \mathbf{\Gamma}_1^{-1} \mathbf{A}_1 \mathbf{w}_1$

Stage 2: The index uses economic weights directly: $I_2 = \mathbf{w}'\gamma$

Adjusted genomic covariance matrix:

$$\mathbf{\Gamma}^* = \mathbf{\Gamma} - u \frac{\mathbf{A}_1' \beta_1 \beta_1' \mathbf{A}_1}{\beta_1' \mathbf{\Gamma}_1 \beta_1}$$

Adjusted genotypic covariance matrix:

$$\mathbf{C}^* = \mathbf{C} - u \frac{\mathbf{G}_1' \mathbf{b}_1 \mathbf{b}_1' \mathbf{G}_1}{\mathbf{b}_1' \mathbf{P}_1 \mathbf{b}_1}$$

where $u = k_1(k_1 - \tau)$

Accuracy at stage 1: $\rho_{HI_1} = \sqrt{\frac{\beta_1' \mathbf{\Gamma}_1 \beta_1}{\mathbf{w}' \mathbf{C} \mathbf{w}}}$

Accuracy at stage 2: $\rho_{HI_2} = \sqrt{\frac{\mathbf{w}' \mathbf{\Gamma}^* \mathbf{w}}{\mathbf{w}' \mathbf{C}^* \mathbf{w}}}$

Value

List with components:

- beta1 - Stage 1 genomic index coefficients
- w - Economic weights (stage 2 coefficients)
- stage1_metrics - List with stage 1 metrics (R1, E1, rho_HI1)
- stage2_metrics - List with stage 2 metrics (R2, E2, rho_HI2)
- Gamma_star - Adjusted genomic covariance matrix at stage 2

- `C_star` - Adjusted genotypic covariance matrix at stage 2
- `rho_I1I2` - Correlation between stage 1 and stage 2 indices
- `k1` - Selection intensity at stage 1
- `k2` - Selection intensity at stage 2
- `summary_stage1` - Data frame with stage 1 summary
- `summary_stage2` - Data frame with stage 2 summary

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapter 9, Section 9.4.

Examples

```
## Not run:
# Two-stage genomic selection example
# Stage 1: Select based on GEBVs for 3 traits
# Stage 2: Select based on GEBVs for all 7 traits

# Compute covariance matrices
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate GEBV covariances (in practice, compute from genomic prediction)
set.seed(123)
reliability <- 0.7
Gamma1 <- reliability * gmat[1:3, 1:3]
Gamma <- reliability * gmat
A1 <- reliability * gmat[1:3, 1:3]
A <- gmat[, 1:3]

# Economic weights
weights <- c(10, 8, 6, 4, 3, 2, 1)

# Run MLGSI
result <- mlgsi(
  Gamma1 = Gamma1, Gamma = Gamma, A1 = A1, A = A,
  C = gmat, G1 = gmat[1:3, 1:3], P1 = pmat[1:3, 1:3],
  wmat = weights, selection_proportion = 0.1
)

print(result$summary_stage1)
print(result$summary_stage2)

## End(Not run)
```

`mlpsi` *Multistage Linear Phenotypic Selection Index (MLPSI)*

Description

Implements the two-stage Linear Phenotypic Selection Index where selection occurs at two different stages with covariance adjustments due to selection effects using Cochran/Cunningham's method.

Usage

```
mlpsi(
  P1,
  P,
  G1,
  C,
  wmat,
  wcol = 1,
  stage1_indices = NULL,
  selection_proportion = 0.1,
  use_young_method = FALSE,
  k1_manual = 2.063,
  k2_manual = 2.063,
  tau = NULL
)
```

Arguments

`P1`	Phenotypic variance-covariance matrix for stage 1 traits (n1 x n1)
`P`	Phenotypic variance-covariance matrix for all traits at stage 2 (n x n)
`G1`	Genotypic variance-covariance matrix for stage 1 traits (n1 x n1)
`C`	Genotypic variance-covariance matrix for all traits (n x n)
`wmat`	Economic weights vector or matrix (n x k)
`wcol`	Weight column to use if wmat has multiple columns (default: 1)
`stage1_indices`	Integer vector specifying which traits (columns of P and C) correspond to stage 1. Default is 1:nrow(P1), assuming stage 1 traits are the first n1 traits. Use this to specify non-contiguous traits, e.g., c(1, 3, 5) for traits 1, 3, and 5.
`selection_proportion`	Proportion selected at each stage (default: 0.1)
`use_young_method`	Logical. Use Young's method for selection intensities (default: FALSE). Young's method tends to overestimate intensities; manual intensities are recommended.
`k1_manual`	Manual selection intensity for stage 1 (used if use_young_method = FALSE)
`k2_manual`	Manual selection intensity for stage 2 (used if use_young_method = FALSE)
`tau`	Standardized truncation point (default: computed from selection_proportion)

Details

Mathematical Formulation:

Stage 1 index coefficients:

$$\mathbf{b}_1 = \mathbf{P}_1^{-1}\mathbf{G}_1\mathbf{w}$$

Stage 2 index coefficients:

$$\mathbf{b}_2 = \mathbf{P}^{-1}\mathbf{G}\mathbf{w}$$

Adjusted phenotypic covariance matrix (Cochran/Cunningham):

$$\mathbf{P}^* = \mathbf{P} - u\frac{Cov(\mathbf{y}, \mathbf{x}_1)\mathbf{b}_1\mathbf{b}_1' Cov(\mathbf{x}_1, \mathbf{y})}{\mathbf{b}_1'\mathbf{P}_1\mathbf{b}_1}$$

Adjusted genotypic covariance matrix:

$$\mathbf{C}^* = \mathbf{C} - u\frac{\mathbf{G}_1'\mathbf{b}_1\mathbf{b}_1'\mathbf{G}_1}{\mathbf{b}_1'\mathbf{P}_1\mathbf{b}_1}$$

where $u = k_1(k_1 - \tau)$

Selection response: $R_1 = k_1\sqrt{\mathbf{b}_1'\mathbf{P}_1\mathbf{b}_1}$, $R_2 = k_2\sqrt{\mathbf{b}_2'\mathbf{P}^*\mathbf{b}_2}$

Value

List with components:

- `b1` - Stage 1 index coefficients
- `b2` - Stage 2 index coefficients
- `stage1_metrics` - List with stage 1 metrics (R1, E1, rho_H1)
- `stage2_metrics` - List with stage 2 metrics (R2, E2, rho_H2)
- `P_star` - Adjusted phenotypic covariance matrix at stage 2
- `C_star` - Adjusted genotypic covariance matrix at stage 2
- `rho_12` - Correlation between stage 1 and stage 2 indices
- `k1` - Selection intensity at stage 1
- `k2` - Selection intensity at stage 2
- `summary_stage1` - Data frame with stage 1 summary
- `summary_stage2` - Data frame with stage 2 summary

References

Cochran, W. G. (1951). Improvement by means of selection. Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 449-470.

Cunningham, E. P. (1975). Multi-stage index selection. Theoretical and Applied Genetics, 46(2), 55-61.

Young, S. S. Y. (1964). Multi-stage selection for genetic gain. Heredity, 19(1), 131-144.

Examples

```
## Not run:
# Two-stage selection example
# Stage 1: Select based on 3 traits
# Stage 2: Select based on all 7 traits

# Compute variance-covariance matrices
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Stage 1 uses first 3 traits
P1 <- pmat[1:3, 1:3]
G1 <- gmat[1:3, 1:3]

# Stage 2 uses all 7 traits
P <- pmat
C <- gmat

# Economic weights
weights <- c(10, 8, 6, 4, 3, 2, 1)

# Run MLPSI (default: stage1_indices = 1:3)
result <- mlpsi(
  P1 = P1, P = P, G1 = G1, C = C, wmat = weights,
  selection_proportion = 0.1
)

# Or with non-contiguous traits (e.g., traits 1, 3, 5 at stage 1):
# P1 <- pmat[c(1,3,5), c(1,3,5)]
# G1 <- gmat[c(1,3,5), c(1,3,5)]
# result <- mlpsi(P1 = P1, P = P, G1 = G1, C = C, wmat = weights,
#                 stage1_indices = c(1, 3, 5))

print(result$summary_stage1)
print(result$summary_stage2)

## End(Not run)
```

`mppg_lgsi` *Multistage Predetermined Proportional Gain Linear Genomic Selection Index (MPPG-LGSI)*

Description

Implements the two-stage Predetermined Proportional Gain LGSI where breeders specify desired proportional gains between traits at each stage using GEBVs.

Usage

```
mppg_lgsi(
  Gamma1,
  Gamma,
  A1,
  A,
  C,
  G1,
  P1,
  wmat,
  wcol = 1,
  d1,
  d2,
  U1 = NULL,
  U2 = NULL,
  selection_proportion = 0.1,
  use_young_method = FALSE,
  k1_manual = 2.063,
  k2_manual = 2.063,
  tau = NULL
)
```

Arguments

Gamma1	GEBV variance-covariance matrix for stage 1 traits (n1 x n1)
Gamma	GEBV variance-covariance matrix for all traits at stage 2 (n x n)
A1	Covariance matrix between GEBVs and true breeding values for stage 1 (n1 x n1)
A	Covariance matrix between GEBVs and true breeding values for stage 2 (n x n1)
C	Genotypic variance-covariance matrix for all traits (n x n)
G1	Genotypic variance-covariance matrix for stage 1 traits (n1 x n1)
P1	Phenotypic variance-covariance matrix for stage 1 traits (n1 x n1)
wmat	Economic weights vector or matrix (n x k)
wcol	Weight column to use if wmat has multiple columns (default: 1)
d1	Vector of desired proportional gains for stage 1 (length n1)
d2	Vector of desired proportional gains for stage 2 (length n)
U1	Constraint matrix for stage 1 (n1 x r1), optional
U2	Constraint matrix for stage 2 (n x r2), optional
selection_proportion	Proportion selected at each stage (default: 0.1)
use_young_method	Logical. Use Young's method for selection intensities (default: FALSE). Young's method tends to overestimate intensities; manual intensities are recommended.
k1_manual	Manual selection intensity for stage 1

k2_manual	Manual selection intensity for stage 2
tau	Standardized truncation point

Details

Mathematical Formulation:

The PPG genomic coefficients are:

$$\beta_{P_1} = \beta_{R_1} + \theta_1 \mathbf{U}_1 (\mathbf{U}_1' \mathbf{\Gamma}_1 \mathbf{U}_1)^{-1} \mathbf{d}_1$$

$$\beta_{P_2} = \beta_{R_2} + \theta_2 \mathbf{U}_2 (\mathbf{U}_2' \mathbf{\Gamma} \mathbf{U}_2)^{-1} \mathbf{d}_2$$

where proportionality constants are:

$$\theta_1 = \frac{\mathbf{d}_1' (\mathbf{U}_1' \mathbf{\Gamma}_1 \mathbf{U}_1)^{-1} \mathbf{U}_1' \mathbf{A}_1 \mathbf{w}}{\mathbf{d}_1' (\mathbf{U}_1' \mathbf{\Gamma}_1 \mathbf{U}_1)^{-1} \mathbf{d}_1}$$

Covariance Adjustment:

The genetic covariance matrix $\mathbf{C}^*$ is adjusted using phenotypic PPG coefficients $\mathbf{b}_{P1} = \mathbf{P}_1^{-1} \mathbf{G}_1 \mathbf{P}_1^{-1} \mathbf{d}_1$, which reflect the same proportional gain constraints as the genomic coefficients β_{P1}. This ensures the adjustment reflects the actual PPG selection occurring at stage 1.

Important: When using custom `U1` matrices (subset constraints), the phenotypic proxy $\mathbf{b}_{P1}$ uses the standard Tallis formula (all traits constrained), while the genomic index β_{P1} respects the `U1` subset. This may cause $\mathbf{C}^*$ to be slightly over-adjusted. For exact adjustment, use `U1 = NULL` (default, all traits constrained). Calculating the exact restricted phenotypic proxy would require implementing the full MPPG-LPSI projection matrix method for the phenotypic coefficients.

Note: Input covariance matrices (`C`, `P1`, `Gamma1`, `Gamma`) should be positive definite. Non-positive definite matrices may lead to invalid results or warnings.

Value

List with components similar to mlgsi, plus:

- `beta_P1` - PPG genomic stage 1 coefficients
- `beta_P2` - PPG genomic stage 2 coefficients
- `b_P1` - PPG phenotypic stage 1 coefficients (used for C* adjustment)
- `theta1` - Proportionality constant for stage 1
- `theta2` - Proportionality constant for stage 2
- `gain_ratios_1` - Achieved gain ratios at stage 1
- `gain_ratios_2` - Achieved gain ratios at stage 2

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapter 9, Section 9.6.

Examples

```
## Not run:
# Two-stage proportional gain genomic selection
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

reliability <- 0.7
Gamma1 <- reliability * gmat[1:3, 1:3]
Gamma <- reliability * gmat
A1 <- reliability * gmat[1:3, 1:3]
A <- gmat[, 1:3]

# Desired proportional gains
d1 <- c(2, 1, 1)
d2 <- c(3, 2, 1, 1, 1, 0.5, 0.5)

weights <- c(10, 8, 6, 4, 3, 2, 1)

result <- mppg_lgsi(
  Gamma1 = Gamma1, Gamma = Gamma, A1 = A1, A = A,
  C = gmat, G1 = gmat[1:3, 1:3], P1 = pmat[1:3, 1:3],
  wmat = weights, d1 = d1, d2 = d2
)

## End(Not run)
```

---

mppg_lpsi *Multistage Predetermined Proportional Gain Linear Phenotypic Selection Index (MPPG-LPSI)*

Description

Implements the two-stage Predetermined Proportional Gain LPSI where breeders specify desired proportional gains between traits at each stage.

Usage

```
mppg_lpsi(
  P1,
  P,
  G1,
  C,
  wmat,
  wcol = 1,
  d1,
  d2,
  stage1_indices = NULL,
  selection_proportion = 0.1,
```

```
  use_young_method = FALSE,
  k1_manual = 2.063,
  k2_manual = 2.063,
  tau = NULL
)
```

Arguments

P1	Phenotypic variance-covariance matrix for stage 1 traits (n1 x n1)
P	Phenotypic variance-covariance matrix for all traits at stage 2 (n x n)
G1	Genotypic variance-covariance matrix for stage 1 traits (n1 x n1)
C	Genotypic variance-covariance matrix for all traits (n x n)
wmat	Economic weights vector or matrix (n x k)
wcol	Weight column to use if wmat has multiple columns (default: 1)
d1	Vector of desired proportional gains for stage 1 (length n1)
d2	Vector of desired proportional gains for stage 2 (length n)
stage1_indices	Integer vector specifying which traits correspond to stage 1 (default: 1:nrow(P1))
selection_proportion	Proportion selected at each stage (default: 0.1)
use_young_method	Logical. Use Young's method for selection intensities (default: FALSE). Young's method tends to overestimate intensities; manual intensities are recommended.
k1_manual	Manual selection intensity for stage 1
k2_manual	Manual selection intensity for stage 2
tau	Standardized truncation point

Details

Mathematical Formulation (Chapter 9.3.1, Eq 9.17):

The PPG coefficients are computed using the projection matrix method:

$$\mathbf{b}_{M_1} = \mathbf{b}_{R_1} + \theta_1 \mathbf{U}_1 (\mathbf{U}_1' \mathbf{G}_1 \mathbf{P}_1^{-1} \mathbf{G}_1 \mathbf{U}_1)^{-1} \mathbf{d}_1$$

$$\mathbf{b}_{M_2} = \mathbf{b}_{R_2} + \theta_2 \mathbf{U}_2 (\mathbf{U}_2' \mathbf{C} \mathbf{P}^{-1} \mathbf{C} \mathbf{U}_2)^{-1} \mathbf{d}_2$$

where:

- $\mathbf{b}_{R_i} = \mathbf{K}_{M_i} \mathbf{b}_i$ are restricted coefficients
- $\mathbf{K}_{M_i} = \mathbf{I} - \mathbf{Q}_{M_i}$ is the projection matrix
- θ_i is the proportionality constant computed from $\mathbf{d}_i$
- $\mathbf{U}_i = \mathbf{I}$ (all traits constrained)

$$\mathbf{b}_{M_1} = \mathbf{K}_{M_1} \mathbf{b}_1$$

$$\mathbf{b}_{M_2} = \mathbf{K}_{M_2} \mathbf{b}_2$$

where $\mathbf{K}_{M_i}$ is computed to achieve proportional gains specified by $\mathbf{d}_i$

Value

List with components similar to mlpsi, plus:

- b_M1 - PPG stage 1 coefficients
- b_M2 - PPG stage 2 coefficients
- b_R1 - Restricted stage 1 coefficients
- b_R2 - Restricted stage 2 coefficients
- K_M1 - PPG projection matrix for stage 1
- K_M2 - PPG projection matrix for stage 2
- theta1 - Proportionality constant for stage 1
- theta2 - Proportionality constant for stage 2
- gain_ratios_1 - Achieved gain ratios at stage 1
- gain_ratios_2 - Achieved gain ratios at stage 2

References

Tallis, G. M. (1962). A selection index for optimum genotype. Biometrics, 18(1), 120-122.

Examples

```
## Not run:
# Two-stage proportional gain selection
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

P1 <- pmat[1:3, 1:3]
G1 <- gmat[1:3, 1:3]
P <- pmat
C <- gmat

# Desired proportional gains
d1 <- c(2, 1, 1) # Trait 1 gains twice as much at stage 1
d2 <- c(3, 2, 1, 1, 1, 0.5, 0.5) # Different proportions at stage 2

weights <- c(10, 8, 6, 4, 3, 2, 1)

result <- mppg_lpsi(
  P1 = P1, P = P, G1 = G1, C = C, wmat = weights,
  d1 = d1, d2 = d2
)

## End(Not run)
```

mrlgsi *Multistage Restricted Linear Genomic Selection Index (MRLGSI)*

Description

Implements the two-stage Restricted Linear Genomic Selection Index where certain traits are constrained to have zero genetic gain at each stage using GEBVs.

Usage

```
mrlgsi(
  Gamma1,
  Gamma,
  A1,
  A,
  C,
  G1,
  P1,
  wmat,
  wcol = 1,
  C1,
  C2,
  selection_proportion = 0.1,
  use_young_method = FALSE,
  k1_manual = 2.063,
  k2_manual = 2.063,
  tau = NULL
)
```

Arguments

Gamma1	GEBV variance-covariance matrix for stage 1 traits (n1 x n1)
Gamma	GEBV variance-covariance matrix for all traits at stage 2 (n x n)
A1	Covariance matrix between GEBVs and true breeding values for stage 1 (n1 x n1)
A	Covariance matrix between GEBVs and true breeding values for stage 2 (n x n1)
C	Genotypic variance-covariance matrix for all traits (n x n)
G1	Genotypic variance-covariance matrix for stage 1 traits (n1 x n1)
P1	Phenotypic variance-covariance matrix for stage 1 traits (n1 x n1)
wmat	Economic weights vector or matrix (n x k)
wcol	Weight column to use if wmat has multiple columns (default: 1)
C1	Constraint matrix for stage 1 (n1 x r1)
C2	Constraint matrix for stage 2 (n x r2)

`selection_proportion`
: Proportion selected at each stage (default: 0.1)

`use_young_method`
: Logical. Use Young's method for selection intensities (default: FALSE). Young's method tends to overestimate intensities; manual intensities are recommended.

`k1_manual`
: Manual selection intensity for stage 1

`k2_manual`
: Manual selection intensity for stage 2

`tau`
: Standardized truncation point

Details

Mathematical Formulation:

The restricted genomic coefficients are:

$$\beta_{R_1} = \mathbf{K}_{G_1}\beta_1$$

$$\beta_{R_2} = \mathbf{K}_{G_2}\mathbf{w}$$

where restriction matrices are computed similarly to RLGSI

Value

List with components similar to mlgsi, plus:

- `beta_R1` - Restricted stage 1 coefficients
- `beta_R2` - Restricted stage 2 coefficients
- `K_G1` - Restriction matrix for stage 1
- `K_G2` - Restriction matrix for stage 2

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapter 9, Section 9.5.

Examples

```
## Not run:
# Two-stage restricted genomic selection
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

reliability <- 0.7
Gamma1 <- reliability * gmat[1:3, 1:3]
Gamma <- reliability * gmat
A1 <- reliability * gmat[1:3, 1:3]
A <- gmat[, 1:3]

# Constraint matrices
C1 <- matrix(0, nrow = 3, ncol = 1)
```

```
C1[1, 1] <- 1 # Restrict trait 1 at stage 1

C2 <- matrix(0, nrow = 7, ncol = 2)
C2[1, 1] <- 1 # Restrict trait 1 at stage 2
C2[3, 2] <- 1 # Restrict trait 3 at stage 2

weights <- c(10, 8, 6, 4, 3, 2, 1)

result <- mrlgsi(
  Gamma1 = Gamma1, Gamma = Gamma, A1 = A1, A = A,
  C = gmat, G1 = gmat[1:3, 1:3], P1 = pmat[1:3, 1:3],
  wmat = weights, C1 = C1, C2 = C2
)

## End(Not run)
```

`mrlpsi` *Multistage Restricted Linear Phenotypic Selection Index (MRLPSI)*

Description

Implements the two-stage Restricted Linear Phenotypic Selection Index where certain traits are constrained to have zero genetic gain at each stage.

Usage

```
mrlpsi(
  P1,
  P,
  G1,
  C,
  wmat,
  wcol = 1,
  C1,
  C2,
  stage1_indices = NULL,
  selection_proportion = 0.1,
  use_young_method = FALSE,
  k1_manual = 2.063,
  k2_manual = 2.063,
  tau = NULL
)
```

Arguments

`P1`	Phenotypic variance-covariance matrix for stage 1 traits (n1 x n1)
`P`	Phenotypic variance-covariance matrix for all traits at stage 2 (n x n)
`G1`	Genotypic variance-covariance matrix for stage 1 traits (n1 x n1)

C	Genotypic variance-covariance matrix for all traits (n x n)
wmat	Economic weights vector or matrix (n x k)
wcol	Weight column to use if wmat has multiple columns (default: 1)
C1	Constraint matrix for stage 1 (n1 x r1)
C2	Constraint matrix for stage 2 (n x r2)
stage1_indices	Integer vector specifying which traits correspond to stage 1 (default: 1:nrow(P1))
selection_proportion	Proportion selected at each stage (default: 0.1)
use_young_method	Logical. Use Young's method for selection intensities (default: FALSE). Young's method tends to overestimate intensities; manual intensities are recommended.
k1_manual	Manual selection intensity for stage 1
k2_manual	Manual selection intensity for stage 2
tau	Standardized truncation point

Details

Mathematical Formulation:

The restricted coefficients are computed as:

$$\mathbf{b}_{R_1} = \mathbf{K}_1 \mathbf{b}_1$$

$$\mathbf{b}_{R_2} = \mathbf{K}_2 \mathbf{b}_2$$

where $\mathbf{K}_1 = \mathbf{I}_1 - \mathbf{Q}_1$ and $\mathbf{K}_2 = \mathbf{I}_2 - \mathbf{Q}_2$

and $\mathbf{Q}_i = \mathbf{P}_i^{-1}\mathbf{G}_i\mathbf{C}_i(\mathbf{C}_i'\mathbf{G}_i\mathbf{P}_i^{-1}\mathbf{G}_i\mathbf{C}_i)^{-1}\mathbf{C}_i'\mathbf{G}_i$

Value

List with components similar to mlpsi, plus:

- b_R1 - Restricted stage 1 coefficients
- b_R2 - Restricted stage 2 coefficients
- K1 - Restriction matrix for stage 1
- K2 - Restriction matrix for stage 2

References

Kempthorne, O., & Nordskog, A. W. (1959). Restricted selection indices. Biometrics, 15(1), 10-19.

Examples

```
## Not run:
# Two-stage restricted selection
# Restrict trait 1 at stage 1, traits 1 and 3 at stage 2

pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

P1 <- pmat[1:3, 1:3]
G1 <- gmat[1:3, 1:3]
P <- pmat
C <- gmat

# Constraint matrices
C1 <- matrix(0, nrow = 3, ncol = 1)
C1[1, 1] <- 1 # Restrict trait 1 at stage 1

C2 <- matrix(0, nrow = 7, ncol = 2)
C2[1, 1] <- 1 # Restrict trait 1 at stage 2
C2[3, 2] <- 1 # Restrict trait 3 at stage 2

weights <- c(10, 8, 6, 4, 3, 2, 1)

result <- mrlpsi(
  P1 = P1, P = P, G1 = G1, C = C, wmat = weights,
  C1 = C1, C2 = C2
)

## End(Not run)
```

---

phenomic_genomic_varcov

Phenomic-Genomic Variance-Covariance Matrix (Φ)

Description

Computes the combined phenomic-genomic variance-covariance matrix (Φ or P_L), which is the block matrix representing the joint distribution of phenotypes and GEBVs.

Structure: Φ = [[P, P_yγ], [P_yγ', Γ]]

where: - P = Var(y) = phenotypic variance-covariance - Γ = Var(γ) = genomic variance-covariance - P_yγ = Cov(y, γ) = covariance between phenotypes and GEBVs

Usage

```
phenomic_genomic_varcov(
  phen_mat = NULL,
  gebv_mat = NULL,
  P = NULL,
```

```
  Gamma = NULL,
  P_yg = NULL,
  method = "pearson",
  use = "complete.obs"
)
```

Arguments

phen_mat	Matrix of phenotypes (n_genotypes x n_traits). Optional if P and P_yg are provided.
gebv_mat	Matrix of GEBVs (n_genotypes x n_traits). Optional if Gamma and P_yg are provided.
P	Phenotypic variance-covariance matrix (n_traits x n_traits). Optional if phen_mat is provided.
Gamma	Genomic variance-covariance matrix (n_traits x n_traits). Optional if gebv_mat is provided.
P_yg	Covariance between phenotypes and GEBVs (n_traits x n_traits). Optional if phen_mat and gebv_mat are provided.
method	Character string specifying correlation method: "pearson" (default), "kendall", or "spearman"
use	Character string specifying how to handle missing values: "complete.obs" (default), "pairwise.complete.obs", etc.

Details

The phenomic-genomic covariance matrix is used in: - GESIM (Genomic Eigen Selection Index Method) - Combined phenotypic + genomic selection indices

The matrix is constructed as:

$$\Phi = \begin{bmatrix} P & P_{y\gamma} \\ P'_{y\gamma} & \Gamma \end{bmatrix}$$

where the off-diagonal blocks are transposes, ensuring symmetry.

You can provide either: 1. Raw data: phen_mat + gebv_mat (matrices computed internally) 2. Pre-computed matrices: P + Gamma + P_yg

Value

Symmetric block matrix Φ (2*n_traits x 2*n_traits)

References

Cerón-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Chapter 8.

Examples

```
## Not run:
# Simulate data
set.seed(123)
n_genotypes <- 100
n_traits <- 7
phen_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 15, sd = 3),
  nrow = n_genotypes, ncol = n_traits
)
gebv_mat <- matrix(rnorm(n_genotypes * n_traits, mean = 10, sd = 2),
  nrow = n_genotypes, ncol = n_traits
)

# Compute phenomic-genomic covariance
Phi <- phenomic_genomic_varcov(phen_mat, gebv_mat)
print(dim(Phi)) # Should be 14 x 14 (2 * 7 traits)

## End(Not run)
```

---

phen_varcov *Phenotypic Variance-Covariance Analysis*

Description

Phenotypic Variance-Covariance Analysis

Usage

```
phen_varcov(
  data,
  genotypes,
  replication,
  columns = NULL,
  main_plots = NULL,
  design_type = c("RCBD", "LSD", "SPD"),
  method = c("REML", "Yates", "Healy", "Regression", "Mean", "Bartlett")
)
```

Arguments

data	traits to be analyzed
genotypes	vector containing genotypes/treatments (sub-plot treatments in SPD)
replication	vector containing replication/blocks (RCBD) or rows (LSD)
columns	vector containing columns (required for Latin Square Design only)
main_plots	vector containing main plot treatments (required for Split Plot Design only)
design_type	experimental design type: "RCBD" (default), "LSD" (Latin Square), or "SPD" (Split Plot)

method	Method for missing value imputation: "REML" (default), "Yates", "Healy", "Regression", "Mean", or "Bartlett"

Value

A Phenotypic Variance-Covariance Matrix

Examples

```
# RCBD example
phen_varcov(data = seldata[, 3:9], genotypes = seldata$treat, replication = seldata$rep)

# Latin Square Design example (requires columns parameter)
# phen_varcov(data=lsd_data[,3:7], genotypes=lsd_data$treat,
#            replication=lsd_data$row, columns=lsd_data$col, design_type="LSD")

# Split Plot Design example (requires main_plots parameter)
# phen_varcov(data=spd_data[,3:7], genotypes=spd_data$subplot,
#           replication=spd_data$block, main_plots=spd_data$mainplot, design_type="SPD")
```

---

plot.selection_simulation

Plot Method for Selection Simulation Results

Description

Plot Method for Selection Simulation Results

Usage

```
## S3 method for class 'selection_simulation'
plot(x, trait_index = 1, type = c("mean", "gain"), ...)
```

Arguments

x	Object of class selection_simulation
trait_index	Which trait to plot (default: 1)
type	Type of plot: "mean" for mean genetic value, "gain" for genetic gain per cycle
...	Additional arguments passed to plot

ppg_esim *Predetermined Proportional Gain Eigen Selection Index (PPG-ESIM)*

Description

Extends ESIM by enforcing that genetic gains are proportional to a user-specified vector $\mathbf{d}$: $\Delta\mathbf{G} \propto \mathbf{d}$. A similarity transformation $\boldsymbol{\beta}_P = \mathbf{F}\mathbf{b}_P$ aligns the eigenvector with the breeder's desired direction.

Usage

```
ppg_esim(pmat, gmat, d, selection_intensity = 2.063)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits).
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).
d	Numeric vector of desired proportional gains (length n_traits). The *ratios* among elements define target gain proportions. Direction (positive/negative) must reflect desired improvement direction (positive = increase, negative = decrease).
selection_intensity	Selection intensity constant (default: 2.063).

Details

Restriction structure via the Mallard Matrix (Section 7.3):

The PPG-ESIM restricts the $(t-1)$ directions **orthogonal** to $\mathbf{d}$, forcing the genetic gain vector to be collinear with $\mathbf{d}$.

The Mallard matrix $\mathbf{D}_M$ is $t \times (t-1)$: its columns span the orthogonal complement of $\mathbf{d}$, obtained via QR decomposition of $\mathbf{d}/\|\mathbf{d}\|$:

$$\mathbf{Q}_{QR} = [\hat{d} \mid \mathbf{D}_M], \quad \text{QR}(\hat{d}) \to \mathbf{Q}_{QR} \in \mathbb{R}^{t \times t}$$

With $\boldsymbol{\Psi} = \mathbf{C}$ (full-trait case, $\mathbf{U} = \mathbf{I}_t$):

PPG projection matrix ($t-1$ restrictions):

$$\mathbf{Q}_P = \mathbf{P}^{-1}\boldsymbol{\Psi}\mathbf{D}_M(\mathbf{D}_M'\boldsymbol{\Psi}'\mathbf{P}^{-1}\boldsymbol{\Psi}\mathbf{D}_M)^{-1}\mathbf{D}_M'\boldsymbol{\Psi}'$$

$$\mathbf{K}_P = \mathbf{I}_t - \mathbf{Q}_P \quad \text{(rank 1)}$$

Because $\mathbf{K}_P$ has rank 1 (projects onto the $\mathbf{d}$ subspace), $\mathbf{K}_P\mathbf{P}^{-1}\mathbf{C}$ has exactly one positive eigenvalue and its eigenvector produces $\Delta\mathbf{G} \propto \mathbf{d}$.

PPG eigenproblem (rank-1 system):

$$(\mathbf{K}_P\mathbf{P}^{-1}\mathbf{C} - \lambda_P^2\mathbf{I}_t)\mathbf{b}_P = 0$$

Similarity transform:

$$\boldsymbol{\beta}_P = \mathbf{F}\mathbf{b}_P$$

where $\mathbf{F} = \text{diag}(\text{sign}(\mathbf{d}))$ aligns the eigenvector sign with the breeder's intended improvement direction.

Key response metrics:

$$R_P = k_I \sqrt{\boldsymbol{\beta}_P' \mathbf{P} \boldsymbol{\beta}_P}$$

$$\mathbf{E}_P = k_I \frac{\mathbf{C}\boldsymbol{\beta}_P}{\sqrt{\boldsymbol{\beta}_P' \mathbf{P} \boldsymbol{\beta}_P}}$$

Value

Object of class "ppg_esim", a list with:

`summary` Data frame with beta (transformed b), b (raw), hI2, rHI, sigma_I, Delta_G, and lambda2.

`beta` Named numeric vector of post-transformation PPG-ESIM coefficients $\boldsymbol{\beta}_P = \mathbf{F}\mathbf{b}_P$.

`b` Raw eigenvector b_P before similarity transform.

`Delta_G` Named vector of expected genetic gains per trait.

`sigma_I` Index standard deviation.

`hI2` Index heritability.

`rHI` Index accuracy.

`lambda2` Leading eigenvalue of the PPG restricted eigenproblem.

`F_mat` Diagonal similarity transform matrix F (diag(sign(d))).

`K_P` PPG projection matrix (rank 1: projects onto d subspace).

`D_M` Mallard matrix (t x t-1): orthogonal complement of d, used to construct the (t-1) restrictions.

`desired_gains` Input proportional gains vector d.

`selection_intensity` Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 7.3.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Desired proportional gains: increase all traits proportionally
d <- c(2, 1, 1, 1, 1, 1, 1)
result <- ppg_esim(pmat, gmat, d)
print(result)
summary(result)

## End(Not run)
```

ppg_gesim *Predetermined Proportional Gain Genomic Eigen Selection Index (PPG-GESIM)*

Description

Implements the PPG-GESIM which extends GESIM to enforce that genetic gains are proportional to a user-specified vector d. Combines eigen approach with predetermined gain proportions.

Usage

```
ppg_gesim(pmat, gmat, Gamma, d, selection_intensity = 2.063)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits).
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).
Gamma	Covariance between phenotypes and GEBVs (n_traits x n_traits).
d	Numeric vector of desired proportional gains (length n_traits). The ratios among elements define target gain proportions.
selection_intensity	Selection intensity constant k_I (default: 2.063 for 10% selection).

Details

Eigenproblem (Section 8.5):

$$(\mathbf{T}_{PG} - \lambda_{PG}^2 \mathbf{I}_{2t})\boldsymbol{\beta}_{PG} = 0$$

where:

$$\mathbf{T}_{PG} = \mathbf{K}_{RG}\boldsymbol{\Phi}^{-1}\mathbf{A} + \mathbf{B}$$

$$\mathbf{B} = \boldsymbol{\delta}\boldsymbol{\varphi}'$$

Implied economic weights:

$$\mathbf{w}_{PG} = \mathbf{A}^{-1}[\boldsymbol{\Phi} + \mathbf{Q}'_{PG}\mathbf{A}]\boldsymbol{\beta}_{PG}$$

Selection response:

$$R_{PG} = k_I\sqrt{\boldsymbol{\beta}'_{PG}\boldsymbol{\Phi}\boldsymbol{\beta}_{PG}}$$

Expected genetic gain per trait:

$$\mathbf{E}_{PG} = k_I \frac{\mathbf{A}\boldsymbol{\beta}_{PG}}{\sqrt{\boldsymbol{\beta}'_{PG}\boldsymbol{\Phi}\boldsymbol{\beta}_{PG}}}$$

Value

Object of class "ppg_gesim", a list with:

summary Data frame with coefficients and metrics.

b_y Coefficients for phenotypic data.

b_gamma Coefficients for GEBVs.

b_combined Combined coefficient vector.

E_PG Expected genetic gains per trait.

gain_ratios Ratios of actual to desired gains (should be constant).

d Original desired proportional gains (length t).

d_PG Extended proportional gains (length 2t, includes GEBV targets).

sigma_I Standard deviation of the index.

hI2 Index heritability.

rHI Accuracy.

R_PG Selection response.

lambda2 Leading eigenvalue.

implied_w Implied economic weights.

U_PG Restriction matrix ((2t-1) x 2t).

selection_intensity Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 8.5.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate GEBV covariance
Gamma <- gmat * 0.8

# Desired proportional gains (e.g., 2:1:3 ratio for first 3 traits)
d <- c(2, 1, 3, 1, 1, 1, 1)

result <- ppg_gesim(pmat, gmat, Gamma, d)
print(result)
print(result$gain_ratios) # Should be approximately constant

## End(Not run)
```

ppg_lgsi *Predetermined Proportional Gains Linear Genomic Selection Index (PPG-LGSI)*

Description

Implements the PPG-LGSI where breeders specify desired proportional gains between traits rather than restricting specific traits to zero. This is genomic version of PPG-LPSI using GEBVs only.

Usage

```
ppg_lgsi(
  Gamma,
  d,
  wmat = NULL,
  wcol = 1,
  U = NULL,
  k_I = 2.063,
  L_G = 1,
  gmat = NULL,
  GAY = NULL
)
```

Arguments

Gamma	GEBV variance-covariance matrix (n_traits x n_traits)
d	Vector of desired proportional gains (length n_traits or n_constraints). If length n_traits, constraints are applied to all traits. If length n_constraints, must provide U matrix.
wmat	Optional. Economic weights for GA/PRE calculation
wcol	Weight column to use if wmat has multiple columns (default: 1)
U	Optional. Constraint matrix (n_traits x n_constraints). If NULL, assumes d applies to all traits (U = I).
k_I	Selection intensity (default: 2.063)
L_G	Standardization constant (default: 1)
gmat	Optional. True genetic variance-covariance matrix for exact accuracy calculation. If NULL, uses Gamma as approximation.
GAY	Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation (Chapter 6, Section 6.2):

Alternative form: $beta_P G = beta_R G + theta_G * U * (U' * Gamma * U)^{-1} * d$

Where: - beta_RG = Restricted index coefficients (from RLGSI) - theta_G = Proportionality constant - d = Vector of desired proportional gains

Proportionality constant:

$$theta_G = (d' * (U' * Gamma * U)^{-1} * U' * Gamma * w)/(d' * (U' * Gamma * U)^{-1} * d)$$

Value

List with:

- summary - Data frame with coefficients and metrics
- b - Vector of PPG-LGSI coefficients (β_{PG})
- E - Named vector of expected genetic gains per trait
- theta_G - Proportionality constant
- gain_ratios - Ratios of achieved to desired gains

Examples

```
## Not run:
# Simulate GEBV variance-covariance matrix
set.seed(123)
n_traits <- 5
Gamma <- matrix(rnorm(n_traits^2), n_traits, n_traits)
Gamma <- (Gamma + t(Gamma)) / 2
diag(Gamma) <- abs(diag(Gamma)) + 2

# Desired proportional gains (e.g., 2:1:1:0:0 ratio)
d <- c(2, 1, 1, 0, 0)

# Economic weights
w <- c(10, 8, 6, 4, 2)

result <- ppg_lgsi(Gamma, d, wmat = w)
print(result$summary)
print(result$gain_ratios) # Should be approximately proportional to d

## End(Not run)
```

---

ppg_lpsi *Predetermined Proportional Gains (PPG-LPSI)*

Description

Implements the PPG-LPSI where breeders specify desired proportional gains between traits rather than restricting specific traits to zero. Based on Tallis (1962).

Usage

```
ppg_lpsi(pmat, gmat, k, wmat = NULL, wcol = 1, GAY)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits)
gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
k	Vector of desired proportional gains (length n_traits). Example: k = c(2, 1, 1) means trait 1 should gain twice as much as traits 2 and 3.
wmat	Optional weight matrix for GA/PRE calculation
wcol	Weight column number (default: 1)
GAY	Genetic advance of comparative trait (optional)

Details

Mathematical Formulation (Chapter 3, Section 3.2):

The PPG-LPSI achieves gains in specific proportions: Delta_G = phi*k

Coefficient formula (Tallis, 1962):

$$b = P^{-1}G(G'P^{-1}G)^{-1}k$$

Where: - k = Vector of desired proportions - phi = Proportionality constant (determined by selection intensity and variances)

The constraint ensures Delta_G1:Delta_G2:Delta_G3 = k1:k2:k3

Value

List with:

- summary - Data frame with coefficients and metrics
- b - Vector of PPG-LPSI coefficients
- Delta_G - Expected genetic gains per trait
- phi - Proportionality constant

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Gains in ratio 2:1:1:1:1:1:1
k <- c(2, 1, 1, 1, 1, 1, 1)
result <- ppg_lpsi(pmat, gmat, k)

## End(Not run)
```

predict_selection_score *Predict selection index scores*

Description

Predict selection index scores

Usage

```
predict_selection_score(index_df, data, genotypes)
```

Arguments

index_df	Data frame returned by lpsi()
data	Raw phenotypic data matrix/data frame (observations x traits)
genotypes	Vector of genotype/treatment labels for each observation

Value

Data frame of selection index scores by genotype

Examples

```
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
cindex <- lpsi(ncomb = 1, pmat = pmat, gmat = gmat, wmat = weight[, -1], wcol = 1)
predict_selection_score(cindex, data = seldata[, 3:9], genotypes = seldata[, 2])
```

print.base_index *Print method for Base Index*

Description

Print method for Base Index

Usage

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

Arguments

x	Object of class 'base_index'
...	Additional arguments (unused)

print.dg_lpsi *Print method for Desired Gains Index*

Description

Print method for Desired Gains Index

Usage

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

Arguments

x	Object of class 'dg_lpsi'
...	Additional arguments (unused)

print.esim *Print method for ESIM*

Description

Print method for ESIM

Usage

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

Arguments

x	Object of class 'esim'
...	Additional arguments (unused)

print.gesim *Print method for GESIM*

Description

Print method for GESIM

Usage

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

Arguments

x	Object of class 'gesim'
...	Additional arguments (unused)

print.gw_esim *Print method for GW-ESIM*

Description

Print method for GW-ESIM

Usage

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

Arguments

x	Object of class 'gw_esim'
...	Additional arguments (unused)

print.mesim *Print method for MESIM*

Description

Print method for MESIM

Usage

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

Arguments

x	Object of class 'mesim'
...	Additional arguments (unused)

print.ppg_esim *Print method for PPG-ESIM*

Description

Print method for PPG-ESIM

Usage

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

Arguments

x	Object of class 'ppg_esim'
...	Additional arguments (unused)

print.ppg_gesim *Print method for PPG-GESIM*

Description

Print method for PPG-GESIM

Usage

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

Arguments

x	Object of class 'ppg_gesim'
...	Additional arguments (unused)

print.resim *Print method for RESIM*

Description

Print method for RESIM

Usage

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

Arguments

x	Object of class 'resim'
...	Additional arguments (unused)

print.rgesim *Print method for RGESIM*

Description

Print method for RGESIM

Usage

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

Arguments

x	Object of class 'rgesim'
...	Additional arguments (unused)

print.selection_simulation *Print Method for Selection Simulation Results*

Description

Print Method for Selection Simulation Results

Usage

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

Arguments

x	Object of class selection_simulation
...	Additional arguments (not used)

print.smith_hazel *Print method for Smith-Hazel Index*

Description

Print method for Smith-Hazel Index

Usage

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

Arguments

x	Object of class 'smith_hazel'
...	Additional arguments (unused)

resim *Restricted Linear Phenotypic Eigen Selection Index (RESIM)*

Description

Extends ESIM by imposing null restrictions: genetic gains for r selected traits are forced to zero while the index heritability for the remaining traits is maximised.

Usage

```
resim(
  pmat,
  gmat,
  restricted_traits = NULL,
  U_mat = NULL,
  selection_intensity = 2.063
)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits).
gmat	Genotypic variance-covariance matrix (n_traits x n_traits).
restricted_traits	Integer vector of trait indices to restrict to zero genetic gain. Example: `c(1, 3)` restricts traits 1 and 3. Alternatively supply a custom restriction matrix via `U_mat`.
U_mat	Optional. Restriction matrix (n_traits x r) where each column defines one null restriction ($\mathbf{U}'\mathbf{C}\mathbf{b} = 0$). Ignored if `restricted_traits` is provided.
selection_intensity	Selection intensity constant (default: 2.063).

Details

Projection matrix (Section 7.2):

$$\mathbf{K} = \mathbf{I}_t - \mathbf{P}^{-1}\mathbf{C}\mathbf{U}(\mathbf{U}'\mathbf{C}\mathbf{P}^{-1}\mathbf{C}\mathbf{U})^{-1}\mathbf{U}'\mathbf{C}$$

Restricted eigenproblem:

$$(\mathbf{K}\mathbf{P}^{-1}\mathbf{C} - \lambda_R^2\mathbf{I}_t)\mathbf{b}_R = 0$$

Selection response and genetic gain:

$$R_R = k_I\sqrt{\mathbf{b}_R'\mathbf{P}\mathbf{b}_R}$$

$$\mathbf{E}_R = k_I\frac{\mathbf{C}\mathbf{b}_R}{\sqrt{\mathbf{b}_R'\mathbf{P}\mathbf{b}_R}}$$

Implied economic weights:

$$\mathbf{w}_R = \mathbf{C}^{-1}[\mathbf{P} + \mathbf{Q}_R'\mathbf{C}]\mathbf{b}_R$$

where $\mathbf{Q}_R = \mathbf{I} - \mathbf{K}$.

Value

Object of class `"resim"`, a list with:

`summary` Data frame with b coefficients and key metrics.

`b` Named numeric vector of RESIM coefficients.

`Delta_G` Named vector of expected genetic gains per trait.

`sigma_I` Index standard deviation.

`hI2` Index heritability (leading eigenvalue of KP^(-1)C).

`rHI` Index accuracy.

`lambda2` Leading eigenvalue of the restricted eigenproblem.

`K` Projection matrix used.

`U_mat` Restriction matrix used.

`restricted_traits` Integer vector of restricted trait indices.

`implied_w` Implied economic weights.

`selection_intensity` Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 7.2.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Restrict traits 1 and 3 to zero genetic gain
result <- resim(pmat, gmat, restricted_traits = c(1, 3))
print(result)
summary(result)

## End(Not run)
```

`rgesim` *Restricted Linear Genomic Eigen Selection Index Method (RGESIM)*

Description

Implements the RGESIM which extends GESIM to allow restrictions on genetic gains of certain traits. Uses the eigen approach with Lagrange multipliers.

Usage

```
rgesim(pmat, gmat, Gamma, U_mat, selection_intensity = 2.063)
```

Arguments

`pmat`	Phenotypic variance-covariance matrix (n_traits x n_traits).
`gmat`	Genotypic variance-covariance matrix (n_traits x n_traits).
`Gamma`	Covariance between phenotypes and GEBVs (n_traits x n_traits).
`U_mat`	Restriction matrix (r x n_traits) where r is number of restrictions. Each row specifies a linear combination of traits to be held at zero gain.
`selection_intensity`	Selection intensity constant k_I (default: 2.063 for 10% selection).

Details

Eigenproblem (Section 8.4):

$$(\mathbf{K}_{RG}\mathbf{\Phi}^{-1}\mathbf{A} - \lambda_{RG}^2\mathbf{I}_{2t})\boldsymbol{\beta}_{RG} = 0$$

where:

$$\mathbf{K}_{RG} = \mathbf{I}_{2t} - \mathbf{Q}_{RG}$$

$$\mathbf{Q}_{RG} = \mathbf{\Phi}^{-1}\mathbf{A}\mathbf{U}_G(\mathbf{U}_G'\mathbf{A}\mathbf{\Phi}^{-1}\mathbf{A}\mathbf{U}_G)^{-1}\mathbf{U}_G'\mathbf{A}$$

Implied economic weights:

$$\mathbf{w}_{RG} = \mathbf{A}^{-1}[\mathbf{\Phi} + \mathbf{Q}_{RG}'\mathbf{A}]\boldsymbol{\beta}_{RG}$$

Selection response:

$$R_{RG} = k_I \sqrt{\boldsymbol{\beta}'_{RG} \boldsymbol{\Phi} \boldsymbol{\beta}_{RG}}$$

Expected genetic gain per trait:

$$\mathbf{E}_{RG} = k_I \frac{\mathbf{A}\boldsymbol{\beta}_{RG}}{\sqrt{\boldsymbol{\beta}'_{RG} \boldsymbol{\Phi} \boldsymbol{\beta}_{RG}}}$$

Value

Object of class `"rgesim"`, a list with:

`summary` Data frame with coefficients and metrics.

`b_y` Coefficients for phenotypic data.

`b_gamma` Coefficients for GEBVs.

`b_combined` Combined coefficient vector.

`E_RG` Expected genetic gains per trait.

`constrained_response` U' * E (should be near zero).

`sigma_I` Standard deviation of the index.

`hI2` Index heritability.

`rHI` Accuracy.

`R_RG` Selection response.

`lambda2` Leading eigenvalue.

`implied_w` Implied economic weights.

`K_RG` Projection matrix.

`Q_RG` Constraint projection matrix.

`selection_intensity` Selection intensity used.

References

Ceron-Rojas, J. J., & Crossa, J. (2018). Linear Selection Indices in Modern Plant Breeding. Springer International Publishing. Section 8.4.

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Simulate GEBV covariance
Gamma <- gmat * 0.8

# Restrict first trait to zero gain
# U_mat must be (n_traits x n_restrictions)
n_traits <- nrow(gmat)
U_mat <- matrix(0, n_traits, 1)
```

```
U_mat[1, 1] <- 1 # Restrict trait 1

result <- rgesim(pmat, gmat, Gamma, U_mat)
print(result)
print(result$constrained_response) # Should be near zero

## End(Not run)
```

`rlgsi` *Restricted Linear Genomic Selection Index (RLGSI)*

Description

Implements the Restricted Linear Genomic Selection Index where genetic gains are constrained to zero for specific traits while maximizing gains for others. Uses GEBVs only (no phenotypic data required).

Usage

```
rlgsi(
  Gamma,
  wmat,
  wcol = 1,
  restricted_traits = NULL,
  U = NULL,
  k_I = 2.063,
  L_G = 1,
  gmat = NULL,
  GAY = NULL
)
```

Arguments

`Gamma`	GEBV variance-covariance matrix (n_traits x n_traits). This represents the variance of GEBVs, typically computed from predicted breeding values.
`wmat`	Economic weights matrix (n_traits x k), or vector
`wcol`	Weight column to use if wmat has multiple columns (default: 1)
`restricted_traits`	Vector of trait indices to restrict (default: NULL). Example: c(1, 3) restricts traits 1 and 3 to zero gain.
`U`	Constraint matrix (n_traits x n_constraints). Each column defines a restriction. Alternative to restricted_traits for custom constraints. Ignored if restricted_traits is provided.
`k_I`	Selection intensity (default: 2.063 for 10 percent selection)
`L_G`	Standardization constant (default: 1). Can be set to sqrt(w'Gw) for standardization.

gmat	Optional. True genetic variance-covariance matrix for exact accuracy calculation. If NULL, uses Gamma as approximation. Providing gmat ensures textbook-perfect accuracy metric.
GAY	Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation (Chapter 6, Section 6.1):

The RLGSI minimizes the mean squared difference between the index I = β'γ and the breeding objective H = w'g under the restriction: U'$\Gamma\beta$ = 0

Solution involves solving the augmented system:

$$\begin{bmatrix} \Gamma & \Gamma U \\ U'\Gamma & 0 \end{bmatrix} \begin{bmatrix} \beta \\ v \end{bmatrix} = \begin{bmatrix} \Gamma w \\ 0 \end{bmatrix}$$

Where: - Γ (Gamma) = Var(GEBVs) - GEBV variance-covariance matrix - U = Constraint matrix (each column is a restriction vector) - w = Economic weights - β_{RG} = RLGSI coefficient vector - v = Lagrange multipliers

Selection response: $R_{RG} = (k_I/L_G) * sqrt(beta_R G' * Gamma * beta_R G)$

Expected gains: $E_{RG} = (k_I/L_G) * (Gamma * beta_R G)/sqrt(beta_R G' * Gamma * beta_R G)$

Value

List with:

- summary - Data frame with coefficients, response metrics
- b - Vector of RLGSI coefficients (β_{RG})
- E - Named vector of expected genetic gains per trait
- R - Overall selection response
- U - Constraint matrix used
- constrained_response - Realized gains for constrained traits (should be ~0)

Examples

```
## Not run:
# Simulate GEBV variance-covariance matrix
set.seed(123)
n_traits <- 5
Gamma <- matrix(rnorm(n_traits^2), n_traits, n_traits)
Gamma <- (Gamma + t(Gamma)) / 2 # Make symmetric
diag(Gamma) <- abs(diag(Gamma)) + 2 # Ensure positive definite

# Economic weights
w <- c(10, 8, 6, 4, 2)

# Restrict traits 2 and 4 to zero gain
result <- rlgsi(Gamma, w, restricted_traits = c(2, 4))
print(result$summary)
```

```
print(result$E) # Check that traits 2 and 4 have ~0 gain

## End(Not run)
```

`rlpsi` *Restricted Linear Phenotypic Selection Index (RLPSI)*

Description

Implements the Restricted LPSI where genetic gains are constrained to zero for specific traits while maximizing gains for others. Based on Kempthorne & Nordskog (1959).

Usage

```
rlpsi(pmat, gmat, wmat, wcol = 1, restricted_traits = NULL, C = NULL, GAY)
```

Arguments

`pmat`	Phenotypic variance-covariance matrix (n_traits x n_traits)
`gmat`	Genotypic variance-covariance matrix (n_traits x n_traits)
`wmat`	Weight matrix (n_traits x k), or vector
`wcol`	Weight column number (default: 1)
`restricted_traits`	Vector of trait indices to restrict (default: NULL). If provided, a constraint matrix C is auto-generated to enforce zero gain on these traits. Example: c(1, 3) restricts traits 1 and 3 to zero gain.
`C`	Constraint matrix (n_traits x n_constraints). Each column is a restriction. Alternative to restricted_traits for custom constraints. Ignored if restricted_traits is provided.
`GAY`	Genetic advance of comparative trait (optional)

Details

Mathematical Formulation (Chapter 3, Section 3.1):

The RLPSI minimizes the mean squared difference between I = b'y and H = w'g subject to the restriction: C'Delta_G = 0

Coefficient formula:

$$b_r = [I - P^{-1}GC(C'GP^{-1}GC)^{-1}C'G]P^{-1}Gw$$

Where: - P = Phenotypic variance-covariance matrix - G = Genotypic variance-covariance matrix - C = Constraint matrix (each column enforces one restriction) - w = Economic weights

The constraint C'Delta_G = 0 ensures zero genetic gain for restricted traits.

Value

List with:

- summary - Data frame with coefficients (b.*), GA, PRE, Delta_G, rHI, hI2
- b - Numeric vector of selection index coefficients
- Delta_G - Named vector of realized correlated responses per trait
- C - Constraint matrix used

Examples

```
## Not run:
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
wmat <- weight_mat(weight)

# Easy way: Restrict traits 1 and 3 to zero gain
result <- rlpsi(pmat, gmat, wmat, wcol = 1, restricted_traits = c(1, 3))

# Advanced way: Provide custom constraint matrix
C <- diag(ncol(pmat))[, 1, drop = FALSE]
result <- rlpsi(pmat, gmat, wmat, wcol = 1, C = C)

## End(Not run)
```

---

seldata *Selection Index DataSet*

Description

A dataset containing the data of three replications and 48 progenies with 7 different traits.

Usage

```
data(seldata)
```

Format

A data frame with 75 rows and 9 columns

Details

- rep. Replications
- treat. Treatments/Genotypes
- sypp. Seed Yield per Plant
- dtf. Days to 50
- rpp. Racemes per Plant

- ppr. Pods per Raceme
- ppp. Pods per Plant
- spp. Seeds per Pod
- pw. Pods Weight

`simulate_selection_cycles` *Simulate Multi-Cycle Selection Using Different Indices*

Description

Performs a stochastic simulation comparing the long-term performance of four selection indices (LPSI, ESIM, RLPSI, RESIM) over multiple breeding cycles. Models linkage between loci using Haldane's mapping function.

Usage

```
simulate_selection_cycles(
  n_cycles = 50,
  n_individuals = 1000,
  n_loci = 100,
  n_traits = 3,
  heritability = 0.5,
  selection_proportion = 0.1,
  economic_weights = NULL,
  restricted_traits = NULL,
  genetic_distances = NULL,
  qtl_effects = NULL,
  seed = NULL
)
```

Arguments

`n_cycles`	Number of selection cycles to simulate (default: 50)
`n_individuals`	Initial population size (default: 1000)
`n_loci`	Number of QTL per trait (default: 100)
`n_traits`	Number of quantitative traits (default: 3)
`heritability`	Heritability for all traits (scalar or vector, default: 0.5)
`selection_proportion`	Proportion of individuals selected (default: 0.1)
`economic_weights`	Economic weights for LPSI (default: equal weights)
`restricted_traits`	Trait indices to restrict to zero gain for RLPSI/RESIM (default: NULL)

`genetic_distances`
: Vector of genetic distances between loci in Morgans (default: 0.1)

`qtl_effects`
: Optional matrix of QTL effects (n_loci x n_traits)

`seed`
: Random seed for reproducibility (default: NULL)

Details

Simulation Procedure (Chapter 10):

1. Initialize population with random QTL alleles at linked loci 2. For each cycle: - Compute genetic values from diploid genotypes - Add environmental noise to create phenotypes - Calculate variance-covariance matrices - Apply each selection index (LPSI, ESIM, RLPSI, RESIM) - Select top individuals based on index scores - Generate offspring through recombination (using Haldane's function) 3. Track genetic gain and population mean across cycles

The Four Indices (Chapter 10, Section 10.2):

- **LPSI**: Unrestricted index maximizing genetic gain: b = P^(-1)Gw
- **ESIM**: Unrestricted eigen index maximizing accuracy: (P^(-1)C - lambda^2 I)b = 0
- **RLPSI**: Restricted LPSI with constraints: U'Gb = 0
- **RESIM**: Restricted ESIM with constraints: U'Cb = 0

Important Simulation Assumptions:

Note: Environmental variance is calculated once at generation 0 and held constant across all cycles. Heritability will naturally decline over cycles as genetic variance is depleted (Bulmer effect). This models the biological reality that selection exhausts additive genetic variance while environmental variation remains stable.

Value

List with components:

- `lpsi_gain` - Matrix of genetic gains per cycle for LPSI (n_cycles x n_traits)
- `esim_gain` - Matrix of genetic gains per cycle for ESIM
- `rlpsi_gain` - Matrix of genetic gains per cycle for RLPSI
- `resim_gain` - Matrix of genetic gains per cycle for RESIM
- `lpsi_mean` - Mean genetic value per cycle for LPSI (n_cycles x n_traits)
- `esim_mean` - Mean genetic value per cycle for ESIM
- `rlpsi_mean` - Mean genetic value per cycle for RLPSI
- `resim_mean` - Mean genetic value per cycle for RESIM
- `parameters` - List of simulation parameters used

Examples

```
## Not run:
# Basic simulation with 3 traits over 20 cycles
results <- simulate_selection_cycles(
  n_cycles = 20,
  n_individuals = 500,
  n_loci = 50,
  n_traits = 3,
  heritability = 0.5,
  selection_proportion = 0.1,
  economic_weights = c(10, 5, 3),
  seed = 123
)

# Plot genetic gains
plot(1:20, results$lpsi_mean[, 1],
  type = "l", col = "blue",
  ylab = "Mean Genetic Value", xlab = "Cycle",
  main = "Genetic Gain - Trait 1"
)
lines(1:20, results$esim_mean[, 1], col = "red")
lines(1:20, results$rlpsi_mean[, 1], col = "green")
lines(1:20, results$resim_mean[, 1], col = "orange")
legend("topleft", c("LPSI", "ESIM", "RLPSI", "RESIM"),
  col = c("blue", "red", "green", "orange"), lty = 1
)

# Restrict trait 2 to zero gain
results_restricted <- simulate_selection_cycles(
  n_cycles = 20,
  n_traits = 3,
  restricted_traits = 2,
  seed = 456
)

## End(Not run)
```

`smith_hazel` *Smith-Hazel Linear Phenotypic Selection Index*

Description

Implements the optimal Smith-Hazel selection index which maximizes the correlation between the index I = b'y and the breeding objective H = w'g.

This is the foundational selection index method from Chapter 2.

Usage

```
smith_hazel(
```

```
  pmat,
  gmat,
  wmat,
  wcol = 1,
  selection_intensity = 2.063,
  GAY = NULL
)
```

Arguments

pmat	Phenotypic variance-covariance matrix (n_traits x n_traits)
gmat	Genotypic variance-covariance matrix (n_traits x n_traits)
wmat	Economic weights matrix (n_traits x k), or vector
wcol	Weight column to use if wmat has multiple columns (default: 1)
selection_intensity	Selection intensity constant (default: 2.063 for 10% selection)
GAY	Optional. Genetic advance of comparative trait for PRE calculation

Details

Mathematical Formulation (Chapter 2):

Index coefficients: $b = P^{-1}Gw$

Where: - P = Phenotypic variance-covariance matrix - G = Genotypic variance-covariance matrix - w = Economic weights

Key metrics: - Variance of index: $\sigma_I^2 = b'Pb$ - Total genetic advance: $R_H = i\sqrt{b'Pb}$ - Expected gains per trait: $\Delta G = (i/\sigma_I)Gb$ - Heritability of index: $h_I^2 = b'Gb/b'Pb$ - Accuracy: $r_{HI} = \sqrt{b'Gb/b'Pb}$

Value

List with:

- summary - Data frame with coefficients and metrics
- b - Vector of Smith-Hazel index coefficients
- w - Named vector of economic weights
- Delta_G - Named vector of expected genetic gains per trait
- sigma_I - Standard deviation of the index
- GA - Total genetic advance
- PRE - Percent relative efficiency
- hI2 - Heritability of the index
- rHI - Accuracy (correlation with breeding objective)

Examples

```
## Not run:
# Calculate variance-covariance matrices
gmat <- gen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])
pmat <- phen_varcov(seldata[, 3:9], seldata[, 2], seldata[, 1])

# Define economic weights
weights <- c(10, 8, 6, 4, 2, 1, 1)

# Build Smith-Hazel index
result <- smith_hazel(pmat, gmat, weights)
print(result)
summary(result)

## End(Not run)
```

---

summary.base_index *Summary method for Base Index*

Description

Summary method for Base Index

Usage

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

Arguments

object	Object of class 'base_index'
...	Additional arguments passed to print

---

summary.dg_lpsi *Summary method for Desired Gains Index*

Description

Summary method for Desired Gains Index

Usage

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

Arguments

object	Object of class 'dg_lpsi'
...	Additional arguments (unused)

---

summary.esim *Summary method for ESIM*

Description

Summary method for ESIM

Usage

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

Arguments

object	Object of class 'esim'
...	Additional arguments (unused)

---

summary.gesim *Summary method for GESIM*

Description

Summary method for GESIM

Usage

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

Arguments

object	Object of class 'gesim'
...	Additional arguments (unused)

summary.gw_esim *Summary method for GW-ESIM*

Description

Summary method for GW-ESIM

Usage

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

Arguments

object	Object of class 'gw_esim'
...	Additional arguments (unused)

summary.mesim *Summary method for MESIM*

Description

Summary method for MESIM

Usage

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

Arguments

object	Object of class 'mesim'
...	Additional arguments (unused)

summary.ppg_esim *Summary method for PPG-ESIM*

Description

Summary method for PPG-ESIM

Usage

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

Arguments

object	Object of class 'ppg_esim'
...	Additional arguments (unused)

summary.ppg_gesim *Summary method for PPG-GESIM*

Description

Summary method for PPG-GESIM

Usage

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

Arguments

object	Object of class 'ppg_gesim'
...	Additional arguments (unused)

summary.resim *Summary method for RESIM*

Description

Summary method for RESIM

Usage

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

Arguments

object	Object of class 'resim'
...	Additional arguments (unused)

summary.rgesim *Summary method for RGESIM*

Description

Summary method for RGESIM

Usage

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

Arguments

object	Object of class 'rgesim'
...	Additional arguments (unused)

summary.selection_simulation *Summary Method for Selection Simulation Results*

Description

Summary Method for Selection Simulation Results

Usage

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

Arguments

object	Object of class selection_simulation
...	Additional arguments (not used)

summary.smith_hazel *Summary method for Smith-Hazel Index*

Description

Summary method for Smith-Hazel Index

Usage

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

Arguments

object	Object of class 'smith_hazel'
...	Additional arguments passed to print

weight *Weight dataset*

Description

A dataset containing the data of 2 different weights namely euqal weight and broad sense heritability

Usage

```
data(weight)
```

Format

A data frame with 7 rows and 3 columns

Details

- EW. Equal Weight
- h2. Broad Sense Heritability

weight_mat *Convert dataframe to matrix*

Description

Convert dataframe to matrix

Usage

```
weight_mat(data)
```

Arguments

data	dataframe of weight

Value

A matrix

Examples

```
weight_mat(data = weight)
```

Index