Package ‘selectiongain’

October 9, 2016

Type Package
Title A Tool for Calculation and Optimization of the Expected Gain from Multi-Stage Selection
Version 2.0.591
Date 2016-10-09
Author Xuefei Mi, Jose Marulanda, H. Friedrich Utz, Albrecht E. Melchinger (Project contact person: Melchinger@uni-hohenheim.de )
Maintainer Xuefei Mi <mi_xue_fei@hotmail.com>
Depends R (>= 3.0.0), mvtnorm,parallel
Description Multi-stage selection is practiced in numerous fields of life and social sciences and particularly in breeding. A special characteristic of multi-stage selection is that candidates are evaluated in successive stages with increasing intensity and effort, and only a fraction of the superior candidates is selected and promoted to the next stage. For the optimum design of such selection programs, the selection gain plays a crucial role. It can be calculated by integration of a truncated multivariate normal (MVN) distribution. While mathematical formulas for calculating the selection gain and the variance among selected candidates were developed long time ago, solutions for numerical calculation were not available. This package can also be used for optimizing multi-stage selection programs for a given total budget and different costs of evaluating the candidates in each stage.
License GPL-2
NeedsCompilation no
Repository CRAN
Date/Publication 2016-10-09 17:54:53

R topics documented:

multistagecor .................................................. 2
multistagegain ................................................. 4
multistagegain.each ........................................... 6
multistageoptimum.grid ..................................... 8
multistageoptimum.nlm .................................... 10
multistageoptimum.search ................................. 12
**Description**

This function is used to calculate the \((n+1)\)-dimensional correlation matrix \(\Sigma^*\) of \(y\) and \(X\), where \(y\) is the true value (genotypic value in plant breeding) and \(X = \{X_1, ..., X_n\}\) are the values of \(y\)'s observations or selection indices, which are linear combinations of the values of observation from each selection stage.

In a plant breeding context, it is assumed that the genetic structure of the candidates to be selected are genetically fixed, e.g., potential cultivars, clones, inbred lines or testcross progenies of inbred lines with the same or different testers in all stages.

**Usage**

```r
multistagecor(maseff, VGCAandE, VSCA, VLine, ecoweight, rhop, T, L, M, Rep, index, covtype, detail)
```

**Arguments**

- `maseff` is the efficiency of marker-assisted selection (MAS). The default value is NA, which means there is no MAS. If a value between 0 and 1 is assigned to `maseff`, then the first selection stage will be considered as MAS (Heffner et al., 2010). The value of MAS is recommended to be higher than 0.1 to avoid ill-shaped correlation matrix.

- `VGCAandE` is the vector of variance components of genetic effect, genotype × location interaction, genotype × year interaction, genotype × location × year interaction and the plot error. When `VSCA` is specified, the `VGCAandE` refers to the general combining ability, otherwise it stands for genetic effect. The default value is 1,1,1,1,1. Variances types listed in Longin et al. (2007) can be used. For example, `VGCAandE=“VC2”` will set the value as 1,0.5,0.5,1,2.

- `VSCA` is the vector of variance components for specific combining ability. The default value is 0,0,0,0.

- `VLine` is the vector of variance components for line per se. The default value is 0,0,0,0,0.

- `ecoweight` is the vector of economic weight

- `rhop` is the genetic correlation between line per se performance and GCA

- `T` is the vector of number of testers at each stage. If there is no tester applied in a certain stage, the value at this stage has to be 1.

- `L` is the vector of number of locations at each stage.
**multistagecor**

- **M** is the vector of tester type, i.e., number of unrelated inbred lines combined in a single tester in stage j.
- **Rep** is the vector of number of replications at each stage.
- **index** is the control parameter. If it equals TRUE, the optimum selection index of Longin et al. (2007) will be used in the calculation of correlation matrix without MAS.
- **covtype** is the type of the covariance. Longin’s type (covtype=c("LonginII")) is used by default.
- **detail** is the control parameter to decide if the correlation matrix, optimal selection index and covariance matrix will be returned (=TRUE) or only the correlation matrix (FALSE). The default value is FALSE.

**Value**

The default output is a matrix with dimension n+1 and can be used as input parameter of function multistagegain. When value of detail=TRUE, the correlation matrix, optimal selection index and covariance matrix will be given.

**Note**

no further comment

**Author(s)**

Xuefei Mi

**References**


**See Also**

selectiongain()

**Examples**

# example for calculating correlation matrix without MAS
multistagecor(VGCAandE=c(1,0.5,0.5,1,2),L=c(2,10),T=c(1,1),Rep=c(1,1))

# example for calculating correlation matrix with MAS in the first stage
VCGca=c(0.4,0.2,0.2,0.4,2,0)
VCSca=c(0.2,0.2,0.1,1,0.1,0.2)
corr.matrix = multistagecor (maseff=0.4, VGCAandE=VCGca, 
VSCA=VCSca, T=c(1,1,5), L=c(1,3,8), Rep=c(1,1,1))
multistagegain

*Function for calculating the expected multi-stage selection gain*

**Description**

This is the main function of the package and uses the following equation given by Tallis (1961) for $y$, which is the true genotypic value:

$$\frac{\partial m(t)}{\partial t_0}|_{t=0} = E(X_0 = y) = \frac{1}{\sigma_0} \sum_{k=0}^{n} \rho_{0,k} \phi_1(q_k) \Phi_n(A_{k,s}; R_k)$$

to calculate the expected selection gain defined by Cochran (1951) for given correlation matrix and coordinates of the truncation points.

**Usage**

```
multistagegain(corr, Q, alg, parallel, vg)
```

**Arguments**

- **corr** is the correlation matrix of $y$ and $X$, which is introduced in the function multistagecorr. The correlation matrix must be symmetric and positive-definite. If the estimated correlation matrix is negative-definite, it must be adjusted before using this function. Before starting the calculations, it is recommended to check the correlation matrix.

- **Q** are the coordinates of the truncation points, which are the output of the function multistagetp that we are going to introduce.

- **vg** correspond to the genetic variance or variance of the GCA effects. The value entered here is only used during the last multiplication of the expected selection gain times the squared root of the genetic variance or the variance of the GCA effects. The default value is 1, and in this case the breeder is advised to make the multiplication outside the function, as showed in the example by Mi et al. 2014 page 1415.

- **alg** is used to switch between two algorithms. If alg = GenzBretz(), which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If alg = Miwa(), the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use the Miwa algorithm.

- **parallel** is a logical variable to desired if the multiple cores can be used for computing, by default is FALSE. The users have to notice that assign cores also cost time. So this procedure can only be efficient if the dim > 5.
Details

This function calculates the well-known selection gain \( \Delta G \), which is described by Cochran (1951), for multi-stage selection. For one-stage selection the gain is defined as \( \Delta G = i \delta_y \rho_1 \), where \( i \) is the selection intensity, \( \rho_1 \) is the correlation between the true breeding value, which has variance \( \delta_y^2 \), and the selection index (Utz 1969).

Value

The returned value is the expected gain of selection.

Note

No further notes

Author(s)

Xuefei Mi

References


See Also

No link

Examples

Q=c(0.4308,0.9804,1.8603)
corr=matrix(c(1,0.3508,0.3508,0.4979,0.3508,1,0.3016,0.5630,0.3508,0.3016,1,0.5630,0.4979,0.5630,0.5630,1),nrow=4)

multistagegain(corr=corr, Q=Q)

# value  1.227475

multistagegain.each  Function for calculating the selection gain in each stage

Description
In some situations, the user wants to know the increase of $\Delta G$ in each stage so that it is possible to determine the stage which contributes most to $\Delta G$. This function calculates $\Delta G$ stepwise for each stage.

Usage
multistagegain.each(corr, Q, alg)

Arguments

corr is the correlation matrix of $y$ and $X$, which is introduced in the function multistagecorr. The correlation matrix must be symmetric and positive-definite. If the estimated correlation matrix is negative-definite, it must be adjusted before using this function. Before starting the calculations, it is recommended to check the correlation matrix.

Q are the coordinates of the truncation points, which are the output of the function multistagetp that we are going to introduce.

alg is used to switch between two algorithms. If alg = GenzBretz(), which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If alg = Miwa(), the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use the Miwa algorithm.

Details
This function calculates the well-known selection gain $\Delta G$, which is described by Cochran (1951), for each stage.

Value
The output is given as $(\Delta G_1(y), \Delta G_2(y) - \Delta G_1(y), \Delta G_3(y) - \Delta G_2(y), \ldots)$ where $\Delta G_i(y)$ refers to the total selection gain after the first $i$ stages of selection.
Author(s)
Xuefei Mi

References

See Also
selectiongain()

Examples

# example 1
corr=matrix( c(1, 0.3508,0.3508,0.4979,
           0.3508, 1, 0.3016,0.5630,
           0.3508, 0.3016,1, 0.5630,
           0.4979, 0.5630,0.5630,1),
           nrow=4)
multistagegain.each(Q=c(0.4308,0.9804,1.8603),corr=corr)

# examples 2
alpha1<- 1/24
alpha2<- 1
Q=multistagetp(alpha=c(alpha1,alpha2),corr=corr[2:3,2:3])

corr=matrix( c(1, 0.7071068,0.9354143,
           0.7071068,1, 0.7559289,
           0.9354143,0.7559289,1),
           nrow=3)
multistagegain.each(Q=Q,corr=corr)

multistagegain

Function for optimizing multi-stage selection with grid algorithm for a given correlation matrix

Description
This function is used to calculate the maximum of $\Delta G$ for a given correlation matrix by grid search algorithm.

Usage
multistagegain(corr, Vg, num.grid, width, Budget, CostProd, CostTest,Nf,alg,detail,fig,N.upper, N.lower,alpha.nursery,cost.nursery)

Arguments

Vg is genotypic variance $\delta_y^2$. The default value is 1.
corr is the correlation matrix of $y$ and $X$, which is introduced in the function multistagecorr. The correlation matrix must be symmetric and positive-definite. If the estimated correlation matrix is negative-definite, it must be adjusted before using this function. Before starting the calculations, it is recommended to check the correlation matrix.

num.grid is the number of equally distanced points that divided the axis of $x_1$ into num.grid_i - 1 intervals and there are $\prod_i(num.grid_i)$ grids in a n dimensional hyper cube. If num.grid > N_i, then the number of grid points for the i-th axis is N_i. The default value of it is NA.

width is the width between the equally distanced points. The default value is NA.

Budget contains the value of total budget.

CostProd contains the initial costs of producing or providing a candidate in each stage.

CostTest contains a vector with length n reflecting the cost of evaluating a candidate in the tests performed at stage i, i=1,...,n. The cost might vary in different stages.

Nf is the number of finally selected candidates.

detail is the control parameter to decide if the result of all the grids will be given or only the maximum. The default value is FALSE.

alg is used to switch between two algorithms. If alg = GenzBretz(), which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If alg = Miwa(), the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use Miwa algorithm of this parameter.
fig is the control parameter to decide if a figure of contour plot will be saved in the default folder of R. The default value is FALSE, which means no figure will be saved.

N.upper is the vector of upper limits of number of candidates X.

N.lower is the vector of lower limits of number of candidates X.

alpha.nursery is a value that should be 0<x<1, prelimitery test alpha fraction should be used for the stage 1. it is setted to 1 as default, when no prelimitery test "nursery stage".

cost.nursery is a vector of length two c([cost of producing a DH line],[cost of testing a DH in nursery]). The default value is 0,0.

Details

for the new added to parameters "alpha.nursery" and "cost.nursery" since v2.0.47:

After producing new DH lines, breeders do NOT go directly for a selection stage in the field, neither for genomic selection. Most of the times, they prefer to make a small field experiment (called "nursery") in which all DH lines are observed and discarded for other traits as disease resistance. That means, all DH lines with poor resistance will be discarded. At the end of the nursery stage only certain amount of DH lines (alpha) advance to the first selection stage (phenotypic or genomic). Specially in maize that makes sense, because in experience around 90 percent of the new DH lines are very weak in terms of per se performance what make them not suitable as new hybrid parents. Then, budget should not be used to make genotyping on or testcrossing with them. Only the alpha fraction should be used for entering the stage 1 of the multistageoptimum.search function.

More details are available in the Crop Science and Computational Statistics papers.

Value

If detail = FALSE, the output of this functions is a vector with the optimal number of candidates in each stage (N) and the maximum ∆G. Otherwise, the result for all the grid points, which have been calculated, will be exported as a table.

Note

no further comment

Author(s)

Xuefei Mi, Jose Marulanda

References


See Also

selectiongain()

Examples

corr=matrix(c(1, 0.3508, 0.3508, 0.4979, 0.3508, 1, 0.3016, 0.5630, 0.3508, 0.3016, 1, 0.5630, 0.4979, 0.5630, 0.5630, 1), nrow=4)

Budget=200

multistageoptimum.grid(Vg=1, num.grid=11, corr=corr, Budget=Budget, CostProd=c(0.5, 0, 0), CostTest=c(0.5, 1, 1), NF=5, N.upper=rep(Budget, 3), N.lower=rep(1, 3))

multistageoptimum.grid(Vg=1, num.grid=11, corr=corr, Budget=Budget, CostProd=c(0.5, 0, 0), CostTest=c(0.5, 1, 1), NF=5, N.upper=rep(Budget, 3), N.lower=rep(1, 3), detail=TRUE, fig=TRUE)

---

multistageoptimum.nlm  Function for optimizing n-stage selection with the NLM algorithm for a given correlation matrix

Description

This function is used to calculate the maximum of $\Delta G$ with given correlation matrix by non-linear minimization algorithm.

Usage

multistageoptimum.nlm(corr, Vg, ini.value, Budget, CostProd, CostTest, NF, iterlim, alg, N.upper, N.lower)
Arguments

**corr** is the correlation matrix of y and X, which is introduced in function multistagecorr. The correlation matrix must be symmetric and positive-definite. Before starting the calculations, the user is recommended to check the correlation matrix.

**Vg** is genotypic variance $\delta_y^2$. The default value is 1.

**ini.value** is a vector, which stores the number of candidates in each stage for the algorithm to begin with. As default, it will use $N = \{N_1, N_2, ..., N_n\} = \{a+1, ..., a+n\}$, where $a$ is defined as $(N.upper + N.lower)/4$.

**Budget** contains the value of total budget.

**CostProd** contains the initial costs of producing or providing a candidate in each stage.

**CostTest** contains a vector with length n reflecting the cost of evaluating a candidate in the tests performed at stage $i$, $i=1,...,n$. The cost might vary in different stages.

**Nf** is the number of finally selected candidates.

**iterlim** is the maximum number of iterations to be executed before the Newton algorithm is terminated. By default it is equal to 20. If the **Budget** increases 10 times for making the selection, the value of **iterlim** has to be increased $lg(10)$ times.

**alg** is used to switch between two algorithms. If **alg** = GenzBretz(), which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If **alg** = Miwa(), the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend the user to use Miwa algorithm of this parameter.

**N.upper** is the vector of up limits of number of candidates X.

**N.lower** is the vector of low limits of number of candidates X.

Value

The output of this function is a vector similar as in multistageoptimal.grid(). However, the optimal number of candidates in each stage determined by the NLM algorithm is clearly not an integer, because the function uses a numerical algorithm, which depends on derivatives.

Note

no further comment

Author(s)

Xuefei Mi
References


See Also

selectiongain()

Examples

VCGCAandError=c(0.40,0.20,0.20,0.40,2.00)
VCSA=c(0.20,0.10,0.10,0.20)

corr = multistagecor (maseff=0.40,
   VCGCAandError=VCGCAandError, VSCA=VCSA, T=c(1,1,5),
   L=c(1,3,8), Rep=c(1,1,1))

# the time of nlm have to be controled in 5 s, so this example will not be uploaded into cran

#multistageoptimum.nlm( corr=corr, Vg=0.4,
#Budget=1021, CostProd=c(0.5,0,0),CostTest=c(0.5,6,40), Nf=10,
# N.upper=c(600,120,20), N.lower=rep(5,3))

multistageoptimum.search

Function for optimizing three-stage selection in plant breeding with one marker-assisted selection stage and two phenotypic selection stages

Description

This function is used to calculate the maximum of $\Delta G$ based on correlation matrix, which depends on locations, testers and replicates, with a grid search algorithm. The changing correlation matrix of three-stage selection are the testcross progenies of DH lines in one marker-assisted selection (MAS) stage and two phenotypic selection (PS) stages.
multistageoptimum.search

Usage

```
multistageoptimum.search (maseff=0.4, VGCAandE, 
VSCA, CostProd, CostTest, Nf, Budget, N2grid, 
N3grid, L2grid, L3grid, T2grid, T3grid, R2, R3, alg, 
detail, fig, alpha.nursery, cost.nursery, 
t2free, parallel.search)
```

Arguments

- **maseff** is the efficiency of MAS.
- **VGCAandE** is the vector of variance components of genetic effect, genotype × location interaction, genotype × year interaction, genotype × location × year interaction and the plot error. When VSCA is specified, it refers to the general combining ability, otherwise it stands for genetic effect. The default value is 1,1,1,1,1. Variances types listed in Longin et al. (2007) can be used. E.g., VGCAandE="VC2" will set the value as 1,0.5,0.5,1,2.
- **VSCA** is the vector of variance components for specific combining ability.
- **CostProd** contains the initial costs of producing or identifying a candidate in each stage.
- **CostTest** contains a vector with length n reflecting the cost of evaluating a candidate in the tests performed at stage i, i=1,...,n. The cost might vary in different stages.
- **Nf** is the number of finally selected candidates.
- **Budget** contains the value of total budget.
- **N2grid** is the vector of lower and upper limits as well as the grid width of number of candidates in the first field test stage.
- **N3grid** is the vector of lower and upper limits as well as the grid width of number of candidates in the second field test stage.
- **L2grid** is the vector of lower and upper limits of number of location as well as the width in the first field test stage.
- **L3grid** is the vector of lower and upper limits of number of location as well as the width in the second field test stage.
- **T2grid** is the vector of lower and upper limits of number of tester as well as the width in the first field test stage.
- **T3grid** is the vector of lower and upper limits of number of tester as well as the width in the second field test stage.
- **R2** is the number of replications in the first field test stage. By default it is 1.
- **R3** is the number of replications in the second field test stage. By default it is 1.
- **alg** is used to switch between two algorithms. If `alg = GenzBretz()`, which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If `alg = Miwa()`, the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use the Miwa algorithm.
multistageoptimum.search

**detail**

is the control parameter to decide if the result of all the grids will be given (=TRUE) or only the maximum (=FALSE).

**fig**

is the control parameter to decide if a contour plot will be saved in the default folder of R. The default value is FALSE, which means no figure will be saved.

**alpha.nursery**

a value that should be 0<x<1, preliminetry test alpha fraction should be used for the stage 1. it is setted to 1 as default, when no preliminetry test "nursery stage".

**cost.nursery**

a vector of length two c([cost of producing a DH line],[cost of testing a DH in nursery]). The default value is 0,0.

**t2free**

is a logical value. If =FALSE, the cost of using T3 and T2 testers will be accounted separetaly. If =TRUE, the cost of using T3 and T2 testers will be accounted according to number of testers, i.e., CostProd=c(CostProd[1],CostProd[2]*T2,CostProd[3]*(T3-T2)

**parallel.search**

is a logical variable to desided if the multiple cores can be used for computing, by default is FALSE. The users have to notice that assign cores also cost time. So this procedure can only be efficient if the dim >5.

**Details**

for the new added to parameters "alpha.nursery" and "cost.nursery" since v2.0.47:

After producing new DH lines, breeders do NOT go directly for a selection stage in the field, neither for genomic selection. Most of the times, they prefer to make a small field experiment (called "nursery") in which all DH lines are observed and discarded for other traits as disease resistance. That means, all DH lines with poor resistance will be discarded. At the end of the nursery stage only certain amount of DH lines (alpha) advance to the first selection stage (phenotypic or genomic). Specially in maize that makes sense, because in experience around 90 percent of the new DH lines are very weak in terms of per se performance what make them not suitable as new hybrid parents. Then, budget should not be used to make genotyping on or testcrossing with them. Only the alpha fraction should be used for entering the stage 1 of the multistageoptimum.search function.

More details are available in the Crop Science and Computational Statistics papers.

**Value**

If detail = FALSE, the output of this function is a vector of the optimum allocation i.e., which achieves the maximum $\Delta G$. Otherwise, the result for all the grid points, which have been calculated, will be exported as a table in the Rgui.

**Note**

no further comment

**Author(s)**

Xuefei Mi, Jose Marulanda
References


See Also

selectiongain()

Examples

```R
CostProd = c(0.5, 1, 1)
CostTest = c(0.5, 1, 1)
Budget = 1021
# Budget is very small here to save time in package checking
# for the example in Heffner's paper, please change it to Budget=10021

VCGAandError = c(0.4, 0.2, 0.2, 0.4, 2)
VCSCA = c(0.2, 0.1, 0.1, 0.2)
Nf = 10

multistageoptimum.search (maseff = 0.4, VCGAandE = VCGAandError, VCSCA = VCSCA, CostProd = c(0.5, 1, 1), CostTest = c(0.5, 1, 1), Nf = 10, Budget = Budget, N2grid = c(11, 1211, 30), N3grid = c(11, 211, 5), L2grid = c(1, 11, 1), L3grid = c(6, 6, 1), T2grid = c(1, 2, 1), T3grid = c(3, 5, 1), R2 = 1, R3 = 1, alg = Miwa(),
detail = TRUE, fig = TRUE)
```

Description

This function is used to calculate the maximum of $\Delta G$ based on correlation matrix, which depends on locations, testers and replicates, with a grid search algorithm. The changing correlation matrix of four-stage selection are the testcross progenies of DH lines in one marker-assisted selection (MAS) stage and three phenotypic selection (PS) stages.
Usage

`multistageoptimum.searchThreeS(maseff=0.4, VGCAandE, VSCA, CostProd, CostTest, Nf, Budget, N2grid, N3grid, N4grid, L2grid, L3grid, L4grid, T2grid, T3grid, T4grid, R2, R3, R4, alg, detail, fig, alpha.nursery, cost.nursery, t2free, parallel.search)`

Arguments

- **maseff** is the efficiency of MAS, if set to NA no marker assisted selection or genomic selection is developed in the first stage.
- **VGCAandE** is the vector of variance components of genetic effect, genotype × location interaction, genotype × year interaction, genotype × location × year interaction and the plot error. When VSCA is specified, it refers to the general combining ability, otherwise it stands for genetic effect. The default value is 1,1,1,1,1. Variances types listed in Longin et al. (2007) can be used. E.g., `VGCAandE="VC2"` will set the value as 1,0.5,0.5,1,2.
- **VSCA** is the vector of variance components for specific combining ability.
- **CostProd** contains the initial costs of producing or identifying a candidate in each stage, then the vector should be of length four.
- **CostTest** contains a vector with length n reflecting the cost of evaluating a candidate in the tests performed at stage i, i=1,...,n. The cost might vary in different stages. For this function n=4
- **Nf** is the number of finally selected candidates.
- **Budget** contains the value of total budget.
- **N2grid** is the vector of lower and upper limits as well as the grid width of number of candidates in the first field test stage.
- **N3grid** is the vector of lower and upper limits as well as the grid width of number of candidates in the second field test stage.
- **N4grid** is the vector of lower and upper limits as well as the grid width of number of candidates in the third field test stage.
- **L2grid** is the vector of lower and upper limits of number of location as well as the width in the first field test stage.
- **L3grid** is the vector of lower and upper limits of number of location as well as the width in the second field test stage.
- **L4grid** is the vector of lower and upper limits of number of location as well as the width in the third field test stage.
- **T2grid** is the vector of lower and upper limits of number of tester as well as the width in the first field test stage.
- **T3grid** is the vector of lower and upper limits of number of tester as well as the width in the second field test stage.
- **T4grid** is the vector of lower and upper limits of number of tester as well as the width in the third field test stage.
is the number of replications in the first field test stage. By default it is 1.

R3 is the number of replications in the second field test stage. By default it is 1.

R4 is the number of replications in the third field test stage. By default it is 1.

alg is used to switch between two algorithms. If $\text{alg} = \text{GenzBretz}()$, which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If $\text{alg} = \text{Miwa}()$, the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use the Miwa algorithm.

detail is the control parameter to decide if the result of all the grids will be given (=TRUE) or only the maximum (=FALSE).

fig is the control parameter to decide if a contour plot will be saved in the default folder of R. The default value is FALSE, which means no figure will be saved.

alpha.nursery a value that should be 0<x<1. The alpha fraction, or amount of genotypes preliminarily selected in nurseries, correspond to the fraction entering stage 1 (when MAS is used) or stage 2 (when there is no MAS). It is setted to 1 as default, i.e. no preliminary test “nursery stage”.

cost.nursery a vector of length two c([cost of producing a DH line],[cost of testing a DH in nursery]). The default value is 0,0.

t2free is a logical value. If =FALSE, the cost of using T4, T3 and T2 testers will be accounted seperately. If =TRUE, the cost of using T4, T3 and T2 testers will be accounted according to number of testers, i.e., $\text{Cost}\text{Prod}=c(\text{Cost}\text{Prod}[1],\text{Cost}\text{Prod}[2]*T2,\text{Cost}\text{Prod}[3]*(T3-T2),\text{Cost}\text{Prod}[4] *(T4-T3)$

parallel.search is a logical variable to desided if the multiple cores can be used for computing, by default is FALSE. The users have to notice that assign cores also cost time. So this procedure can only be efficient if the dim >5.

Details

Some breeding programs require more than two phenotypic selection stages. In this programs, a large number of genotypes are assessed for the target trait only in few locations in the first stage and strong selection pressure is applied. The second and third stages of phenotypic selection are developed in a large number of locations including only a reduced number of genotypes. Even if this strategy could lead to a reduced selection gain, it could be of major advantage when breeding programs have biological or operative restrictions to conduct large experiments a in large number of locations. This function allows breeders to estimate the possible increase or reduction of selection gain when moving from two stages of phenotypic selection to three stages and also when a restricted number of genotypes and locations in each of the three stages of phenotypic selection is used.

for the new added to parameters "alpha.nursery" and "cost.nursery" since v2.0.47:

After producing new DH lines, breeders do NOT go directly for a selection stage in the field, neither for genomic selection. Most of the times, they prefer to make a small field experiment (called “nursery”) in which all DH lines are observed and discarded for other traits as disease resistance. That means, all DH lines with poor resistance will be discarded. At the end of the nursery stage
only certain amount of DH lines (alpha) advance to the first selection stage (phenotypic or genomic). Specially in maize that makes sense, because in experience around 90 percent of the new DH lines are very weak in terms of per se performance what make them not suitable as new hybrid parents. Then, budget should not be used to make genotyping on or testcrossing with them. Only the alpha fraction should be used for entering the stage 1 of the multistageoptimum.search function. More details are available in the Crop Science and Computational Statistics papers.

Value

If `detail = FALSE`, the output of this function is a vector of the optimum allocation i.e., which achieves the maximum $\Delta G$. Otherwise, the result for all the grid points, which have been calculated, will be exported as a table in the Rgui.

Note

no further comment

Author(s)

Xuefei Mi, Jose Marulanda

References


See Also

`selectiongain()`

Examples

```r
#VCGCAandError=c(0.4,0.2,0.2,0.4,2)
#VSCA=c(0.2,0.1,0.1,0.2)

# example 1: restriction on the number of location and genotypes
# for phenotypic stages 2 and 3
#multistageoptimum.searchThreeS(maseff=NA, VCGAandE=VCGCAandError, VSCA=VSCA,
#CostProd=c(1,4,4,4), CostTest=c(0,1,1,1), NF=3, Budget=3000,
#N2grid=c(100,500,50), N3grid=c(10,50,5), N4grid=c(10,50,5),
#L2grid=c(1,2,1), L3grid=c(9,10,1), L4grid=c(9,10,1),
#T2grid=c(1,2,1), T3grid=c(5,6,1), T4grid=c(5,6,1),
#R2=1, R3=1, R4=1, alg=Miwa(), detail=FALSE, fig= FALSE, t2free=TRUE)
```
multistagetp

Function for calculating the truncation points

Description

This function calculates the coordinates of the truncation points $Q$ for given selected fractions $\vec{\alpha} = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ and correlation matrix of X. The R function uniroot in core package stats is called internally to solve the truncation point equations.

Usage

```
multistagetp(alpha, corr, alg)
```

Arguments

- **alpha** is probability vector $\vec{\alpha}$ for random variable X. In plant breeding, it is also called the selected fraction.

- **corr** is the correlation matrix of y and X, which is introduced in the function multistagecorr. The correlation matrix must be symmetric and positive-definite. If the estimated correlation matrix is negative-definite, it must be adjusted before using this function. Before starting the calculations, it is recommended to check the correlation matrix.

- **alg** is used to switch between two algorithms. If $\text{alg} = \text{GenzBretz}()$, which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If $\text{alg} = \text{Miwa}()$, the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa’s algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use the Miwa algorithm.

Details

This function calculates the non-equi coordinate quantile vector $Q = \{q_1, q_2, ..., q_n\}$ for a multi-variate normal distribution from a given $\vec{\alpha}$. It can be compared with the function qmvnorm() in R-package mvtnorm, which calculates only the equi coordinate quantile $q$ for multi-variate normal distribution from a given $\vec{\alpha}$. The function multistagetp is used by function multistagegain to calculate the expected gain.
Value

The output is a vector of the coordinates.

Note

When a $\vec{\alpha}$ is given, the quantiles are calculated consecutively to satisfy the given $\vec{\alpha}$. The calculation from other direction to $-\infty$ of the integral is also possible for qmvnorm().

Author(s)

Xuefei Mi

References


See Also

selectiongain(), qnorm()

Examples

# first example
VC기가andError=c(0.40,0.20,0.20,0.40,2.00)
VC기가SA=c(0.20,0.10,0.10,0.20)

corr.matrix = multistagecor(maseff=0.40, VG가AandError=VC기가AandError,
VC기가SA=VC기가SA, T=c(1,1,5), L=c(1,3,8), Rep=c(1,1,1))

N1=4500;N2=919;N3=45;NF=10

Q=multistagetp(c(N2/N1,N3/N2,NF/N3), corr=corr.matrix)
Description

This function uses the algorithm described by Tallis (1961) to calculate the variance after multi-stage selection. The variance among candidates of $y$ in the selected area $S_Q$ is defined as the second central moment, $\psi_n(y) = E(Y^2 | S_Q) - [E(Y | S_Q)]^2$, where $E(Y^2 | S_Q) = \alpha^{-1} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} y^2 \phi_{n+1}(x^*; \Sigma^*) dx^*$

Usage

`multistagevariance(Q, corr, alg)`

Arguments

- **Q** are the coordinates of the truncation points, which are the output of the function `multistagetp` that we are going to introduce.
- **corr** is the correlation matrix of $y$ and $X$, which is introduced in the function `multistagecorr`. The correlation matrix must be symmetric and positive-definite. If the estimated correlation matrix is negative-definite, it must be adjusted before using this function. Before starting the calculations, it is recommended to check the correlation matrix.
- **alg** is used to switch between two algorithms. If `alg = GenzBretz()`, which is by default, the quasi-Monte Carlo algorithm from Genz et al. (2009, 2013), will be used. If `alg = Miwa()`, the program will use the Miwa algorithm (Mi et al., 2009), which is an analytical solution of the MVN integral. Miwa's algorithm has higher accuracy (7 digits) than quasi-Monte Carlo algorithm (5 digits). However, its computational speed is slower. We recommend to use the Miwa algorithm.

Value

The output is the value of $\psi_n(y | S_Q)$.

Note

No further notes

Author(s)

Xuefei Mi
References


See Also

No link

Examples

# first example

Q = c(0.4308, 0.9804, 1.8603)

corr = matrix(c(1, 0.3508, 0.3508, 0.4979,
                0.3508, 1, 0.3016, 0.5630,
                0.3508, 0.3016, 1, 0.5630,
                0.4979, 0.5630, 0.5630, 1),
               nrow=4)

multistagevariance(Q=Q, corr=corr, alg='Miwa')

# time comparison

var.time.miwa = system.time (var.miwa <- multistagevariance(Q=Q, corr=corr, alg='Miwa'))

var.time.bretz = system.time (var.bretz <- multistagevariance(Q=Q, corr=corr))

# second examples

Q = c(0.9674216, 1.6185430)

corr = matrix(c(1, 0.7071068, 0.9354143,
                0.7071068, 1, 0.7559289,
                0.9354143, 0.7559289, 1),
               nrow=3)
multistagevariance(Q=Q, corr=corr, alg=Miwa)

var.time.miwa = system.time (var.miwa <- multistagevariance(Q=Q, corr=corr, alg=Miwa))

var.time.bretz = system.time (var.bretz <- multistagevariance(Q=Q, corr=corr))

# third examples
alpha1 <- 1/(24)^0.5
alpha2 <- 1/(24)^0.5
Q = multistagetrp(alpha = c(alpha1, alpha2), corr = corr)

corr = matrix(c(1, 0.7071068, 0.9354143, 0.7071068, 1, 0.7559289, 0.9354143, 0.7559289, 1), nrow = 3)

multistagevariance(Q=Q, corr=corr, alg=Miwa)
Index

*Topic Optimization
  multistagecor, 2
  multistageoptimum.grid, 8
  multistageoptimum.nlm, 10
  multistageoptimum.search, 12
  multistageoptimum.searchThreeS, 15

*Topic Truncated multivariate normal
  multistagegain, 4
  multistagegain.each, 6
  multistagetp, 19
  multistagevariance, 21

multistagecor, 2
multistagegain, 4
multistagegain.each, 6
multistageoptimum.grid, 8
multistageoptimum.nlm, 10
multistageoptimum.search, 12
multistageoptimum.searchThreeS, 15
multistagetp, 19
multistagevariance, 21