

Package ‘grmsem’

May 8, 2026

Title Genetic-Relationship-Matrix Structural Equation Modelling (GRMSEM)

Version 1.1.0

Maintainer Beate StPourcain <Beate.StPourcain@mpi.nl>

Description Quantitative genetics tool supporting the modelling of multivariate genetic variance structures in quantitative data. It allows fitting different models through multivariate genetic-relationship-matrix (GRM) structural equation modelling (SEM) in unrelated individuals, using a maximum likelihood approach. Specifically, it combines genome-wide genotyping information, as captured by GRMs, with twin-research-based SEM techniques,
St Pourcain et al. (2017) <[doi:10.1016/j.biopsych.2017.09.020](https://doi.org/10.1016/j.biopsych.2017.09.020)>,
Shapland et al. (2020) <[doi:10.1101/2020.08.14.251199](https://doi.org/10.1101/2020.08.14.251199)>.

Depends R (>= 3.5)

Imports msm (>= 1.6), numDeriv, optimParallel, stats, utils

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Suggests bookdown, knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

URL <https://CRAN.R-project.org/package=grmsem>,
<https://gitlab.gwdg.de/beate.stpourcain/grmsem>

BugReports <https://gitlab.gwdg.de/beate.stpourcain/grmsem/-/issues>

Author Beate StPourcain [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-4680-3517>>),
Alexander Klassmann [ctb]

Repository CRAN

Date/Publication 2021-01-29 09:20:08 UTC

Contents

| | |
|----------------|-----------|
| fit.large | 2 |
| G.small | 3 |
| grm.bin.input | 3 |
| grm.input | 4 |
| grmseml.biher | 4 |
| grmseml.fcoher | 5 |
| grmseml.fit | 6 |
| grmseml.stpar | 10 |
| grmseml.var | 11 |
| ph.large | 12 |
| ph.small | 12 |
| Index | 14 |

| | |
|-----------|--|
| fit.large | <i>Prefitted model: large data set</i> |
|-----------|--|

Description

A large quad-variate data set was simulated assuming an underlying Cholesky model, with 5000 observations per trait and high polygenicity (20,000 SNPs per genetic factor). Genetic trait variances were set to 0.30, 0.60, 0.60 and 0.70 respectively, and residual variances to 0.70, 0.40, 0.40 and 0.30. The data set is described in full, including genetic and residual covariances, in the vignette. A 4-variate Cholesky model was fitted to the data as described in the vignette and the output has been saved in fit.large.RData.

Usage

```
fit.large
```

Format

gsem.fit output (a list object)

model.in input parameters

formula fitted formula

model.fit optimisation output

model.out fitted gsem model

VCOV variance covariance matrix

k Number of phenotypes

n Number of observations across all phenotypes

n.obs Number of observations per phenotype

n.ind Number of individuals with at least one phenotype

model gsem model

con constraint
ph.nms phenotype names

G.small *Symmetric GRM data: small data set*

Description

A genetic relationship matrix for a small tri-variate data set was simulated assuming an underlying Cholesky model, with 100 observations per trait and low polygenicity (150 SNPs per genetic factor). Genetic trait variances were set to 0.30, 0.60, 0.60 and 0.70 respectively, and residual variances to 0.70, 0.40, 0.40 and 0.30. The data set is described in full, including genetic and residual covariances, in the vignette. The traits are influenced by three independent genetic factors (A1, A2 and A3, based on a 150 SNPs each) and three independent residual factors (E1, E2 and E3).

Usage

```
G.small
```

Format

Symmetric GRM data frame with 100 rows and 100 columns (observations)

grm.bin.input *grm import function*

Description

This function imports genetic relationship matrices (GRMs) in binary format as e.g. stored by the GCTA software command `gcta64 --grm test --make-grm-bin --out test`

Usage

```
grm.bin.input(file, size = 4)
```

Arguments

file name of the binary grm file. No default.
size byte size used (typically gcta uses 4). See eponymous parameter of function [readBin](#).

Value

object of type matrix
grmsem.bin.input imports a binary GCTA GRM (.bin) file and transforms it into a symmetric matrix

 grm.input

grm import function

Description

This function imports genetic relationship matrices (GRMs) in gz format as e.g. stored by the GCTA software command `gcta64 --grm test --make-grm-gz --out test`

Usage

```
grm.input(file)
```

Arguments

`file` the name of the gz compressed grm file. No default.

Value

`grmsem.input` imports a zipped GCTA GRM (.gz) file and transforms it into a symmetric matrix

grmsem.biher

grmsem bivariate heritability estimation function.

Description

This function estimates the bivariate heritability.

Usage

```
grmsem.biher(ph, grmsem.var.out = NULL)
```

Arguments

`ph` Phenotype file as R dataframe (columns: ≥ 2 phenotypes, rows: `ni` individuals in the same order as `G`). No default.

`grmsem.var.out` A `grmsem.var` object with unstandardised parameters (factor loadings). Default `NULL`.

Details

The `grmsem.biher` function estimates the bivariate heritability (DS, Cholesky, IP and IPC models) from the observed phenotype data and a `grmsem.var` object. All standard errors are derived with the Delta method.

Value

grmseml.biher returns a list object consisting of the following matrices:

| | |
|----------|--|
| VPO | observed phenotypic variance/covariance matrix |
| VA | estimated genetic variance |
| BIHER | estimated bivariate heritability (off-diagonals): VA / VPO |
| BIHER.se | standard error of estimated bivariate heritability |
| BIHER.Z | Z (wald) of estimated bivariate heritability |
| BIHER.p | p (Wald) of estimated bivariate heritability |

Examples

```

#(runtime should be less than one minute)

out <- grmseml.fit(ph.small, G.small, LogL = TRUE, estSE = TRUE)
var.out <- grmseml.var(out)
grmseml.biher(ph.small, var.out)

```

| | |
|----------------|---|
| grmseml.fcoher | <i>grmseml factorial co-heritability and co-environmentality estimation function.</i> |
|----------------|---|

Description

This function estimates factorial co-heritabilities and factorial co-environmentalities.

Usage

```
grmseml.fcoher(grmseml.out = NULL)
```

Arguments

grmseml.out grmseml.fit or grmseml.stpar object. Default NULL.

Details

The grmseml.fcoher function can be used to estimate factorial co-heritabilities and factorial co-environmentalities for models estimating latent variables (Cholesky, IP or IPC models), based on grmseml.fit or grmseml.stpar objects. The factorial co-heritability of a genetic factor m for trait t is the ratio of the genetic variance explained by factor m (A_{mt}) to the total genetic variance (A_t): A_{mt} / A_t . The factorial co-environmentality of a residual factor n for trait t is the ratio of the residual variance explained by factor n (E_{nt}) to the total residual variance (E_t): E_{nt} / E_t . All standard errors are derived with the Delta method.

Value

grmsem.fcoher returns an extended model.out dataframe, fcoher.model.out, with the following columns:

| | |
|-----------|--|
| label | parameter label |
| estimates | estimated parameters |
| gradient | gradient |
| se | SE |
| Z | Z (Wald) of factor loading |
| p | p (Wald) of factor loading |
| Vi | squared factor loading, explained phenotypic variation by the factor |
| Vi.se | SE of squared factor loading |
| FCOHER | factorial co-heritability |
| FCOHER.se | SE of factorial co-heritability |
| FCOHER.Z | Z (wald) of factorial co-heritability |
| FCOHER.se | p (Wald) of factorial co-heritability |
| FCOENV | factorial co-environmentality |
| FCOENV.se | SE of factorial co-environmentality |
| FCOENV.Z | Z (wald) of factorial co-environmentality |
| FCOENV.se | p (Wald) of factorial co-environmentality |

Examples

```

#(runtime should be less than one minute)

out <- grmsem.fit(ph.small, G.small, LogL = TRUE, estSE = TRUE)
grmsem.fcoher(out)

```

grmsem.fit

grmsem model fitting function

Description

This function fits a grmsem model.

Usage

```
grmsemlfit(
  ph,
  G,
  A.v.free = NULL,
  E.v.free = NULL,
  A.v.startval = NULL,
  E.v.startval = NULL,
  LogL = FALSE,
  estSE = FALSE,
  cores = 1,
  model = "Cholesky",
  compl.ph = FALSE,
  printest = FALSE,
  cluster = "PSOCK",
  optim = "optim",
  verbose = FALSE
)
```

Arguments

| | |
|--------------|--|
| ph | phenotype file as R dataframe, even for single vectors (columns: k phenotypes, rows: ni individuals in same order as G). No default. |
| G | GRM matrix as provided by the grm.input or grm.bin.input.R function. Use the same order of individuals as in ph. No default. |
| A.v.free | vector of free parameters for genetic factor loadings (free:1, not-free:0). Default NULL, all parameters are estimated. |
| E.v.free | vector of free parameters for residual factor loadings (free:1, not-free:0). Default NULL, all parameters are estimated. |
| A.v.startval | vector of starting values for genetic factor loadings. Default NULL, all starting values are generated. |
| E.v.startval | vector of starting values for residual factor loadings. Default NULL, all starting values are generated. |
| LogL | estimation of the loglikelihood using optim BFGS (TRUE/FALSE). Default FALSE. |
| estSE | estimation of standard errors by recalculating the Hessian matrix. Default FALSE. |
| cores | number of cores for multi-threaded calculations (numeric). Default 1. |
| model | grmseml model selection. Options: "Cholesky", "IP", "IPC", "DS". Default "Cholesky". |
| compl.ph | listwise complete observations across all phenotypes (all NA are excluded). Default FALSE. |
| printest | print output of the model.fit function including estimates (printest.txt) that can be used as starting values. Default FALSE. |
| cluster | cluster type. Options: "PSOCK", "FORK". Default "PSOCK". |

| | |
|---------|---|
| optim | optimisation function from stats or optimParallel libraries. Options: "optim", "optimParallel". Default "optim". |
| verbose | additional model fit information: (i) phenotype vector, (ii) n of GRM and corresponding I matrix when data are missing, (iii) Hessian if estSE TRUE. Default FALSE. |

Details

grmsem models estimate genetic (A) and residual (E) variance/covariance of quantitative traits (AE model), where E in GRM-based methods can capture both, shared and unique residual influences. The estimation of parameters and their SEs is performed with the function `grmsem.fit`. Specifically, the loglikelihood is estimated with `stats::optim` and the BFGS (Broyden-Fletcher-Goldfarb-Shannon) approach and the variance/covariance matrix of estimated parameters with `numDeriv::hessian`. The statistical significance of estimated parameters is assessed using a Wald test, assuming multivariate normality.

`grmsem.fit` allows fitting different models describing the underlying multivariate genetic architecture of quantitative traits, as captured by a genetic-relationship-matrix (GRM), using structural equation modelling (SEM) techniques and a maximum likelihood approach. The user can fit multiple predefined model structures to the data. A Cholesky decomposition, Independent Pathway, and hybrid Independent Pathway/Cholesky models can be fitted by setting the `model` option to Cholesky, IP or IPC, respectively. In addition, the Cholesky model can be re-parametrised as Direct Symmetric model, estimating genetic and residual covariances directly, using the `model` option DS. Each model can be adapted by the user by setting free parameters (`A.v.free` and `E.v.free` options) and starting values (`A.v.startval` and `E.v.startval` options).

Input parameters are returned as `model.in` list object. Output from the maximum likelihood estimation is also given as list `model.fit` and the fitted grmsem model with estimated parameters and SEs is returned as dataframe `model.out`. The returned `grmsem.fit` object can be used to estimate genetic and residual covariance and correlations (`grmsem.var` function), bivariate heritabilities (`grmsem.biher` function), and factorial co-heritabilities and co-environmentalities (`grmsem.fcoher` function). All estimated parameters of the fitted grmsem model can also be standardised using the function `grmsem.stpar`.

Listwise complete observations can be selected with the option `compl.ph=TRUE`. Otherwise, `grmsem.fit` fits, like GREML, all available data to the model with the default option `compl.ph FALSE`. Using the option `LogL=FALSE`, the user can check the model input parameters and formula without a maximum likelihood estimation. Using the option `estSE=FALSE`, the user can carry out a maximum likelihood estimation without the estimation of SEs for estimated parameters that require calculating the Hessian. Note that `grmsem.fit` should preferably be run in parallel, by setting the `cores` option to the required number of cores.

When `grmsem.fit` is called with `LogL TRUE`, the user will see also the iterations of the `stats::optim` loglikelihood estimation, which are not included in the exported `grmsem.fit` list object.

Value

`grmsem.fit` returns a `grmsem.fit` list object consisting of:

| | |
|-----------------------|-----------------------------------|
| <code>model.in</code> | list of input parameters |
| <code>formula</code> | matrix of the model specification |

model.fit list output of the maximum likelihood estimation, if LogL TRUE
 model.out dataframe of fitted grmsem model with estimated parameters and SEs, if estSE TRUE
 VCOV variance/covariance matrix
 k number of phenotypes
 n total number of observations across all phenotypes
 n.obs number of observations per phenotype
 n.ind number of individuals with at least one phenotype
 model type of grmsem model
 ph.nms vector of phenotype names

model.in list of input parameters:

part a - genetic, e - residual parameters
 label parameter label
 value starting values
 freepar free parameters

model.fit list output of the maximum likelihood estimation:

optimisation output via optim()
 estimates estimated parameters: factor loadings for 'Cholesky', 'IP' and 'IPC' models, but variance components for 'DS'
 LL loglikelihood
 calls optim() calls
 convergence optim() convergence
 message optim() message

model.out data.frame of fitted grmsem model:

label parameter label
 estimates estimated parameters
 gradient gradient
 se SE
 Z Z (Wald)
 p p (Wald)

Examples

```

#Set up a Cholesky model: Model formula and total number of parameters
#ph.small is a trivariate phenotype file for 100 individuals in the same order as G.small
#nrow = 100, ncol = 3
#(runtime should be less than one minute)
out <- grmsem.fit(ph.small, G.small, LogL = FALSE, estSE = FALSE)

#Run a Cholesky model:
out <- grmsem.fit(ph.small, G.small, LogL = TRUE, estSE = TRUE)

```

| | |
|---------------------------|--|
| <code>grmsem.stpar</code> | <i>grmsem standardised parameters function</i> |
|---------------------------|--|

Description

This function estimates standardised parameters for a `grmsem.fit` object.

Usage

```
grmsem.stpar(grmsem.out = NULL)
```

Arguments

`grmsem.out` `grmsem.fit` object as provided by the `grmsem.fit` function. Default `NULL`.

Details

`grmsem.stpar` standardises `grmsem.fit` estimates so that derived or estimated A and E variance components will add up to phenotypic unit variance. The SEs of standardised parameters are derived using the Jacobian matrix.

Value

`grmsem.stpar` returns a list object consisting of:

| | |
|------------------------------|--|
| <code>model.in</code> | list of input parameters |
| <code>stand.model.out</code> | dataframe of fitted <code>grmsem</code> model with standardised parameters and SEs |
| <code>stVCOV</code> | standardised variance/covariance matrix |
| <code>k</code> | number of phenotypes |
| <code>n</code> | total number of observations across all phenotypes |
| <code>model</code> | type of <code>grmsem</code> model |
| <code>ph.nms</code> | vector of phenotype names |

`model.in` list of input parameters:

| | |
|----------------------|--------------------------------------|
| <code>part</code> | a - genetic, e - residual parameters |
| <code>label</code> | parameter label |
| <code>value</code> | starting values |
| <code>freepar</code> | free parameters |

`stand.model.out` data.frame of fitted `grmsem` model with standardised parameters and SEs:

| | |
|------------------------|-----------------------------------|
| <code>label</code> | parameter label |
| <code>estimates</code> | standardised estimated parameters |
| <code>se</code> | SE |
| <code>Z</code> | Z (Wald) |
| <code>p</code> | p (Wald) |

Examples

```

#(runtime should be less than one minute)

out <- grmseml.fit(ph.small, G.small, LogL = TRUE, estSE = TRUE)
stout <- grmseml.stpar(out)
print(stout)

```

| | |
|-------------|---|
| grmseml.var | <i>grmseml variance estimation function</i> |
|-------------|---|

Description

This function estimates genetic and residual variances, and genetic correlations.

Usage

```
grmseml.var(grmseml.out = NULL)
```

Arguments

grmseml.out A grmseml.fit or grmseml.stpar object. Default NULL.

Details

The grmseml.var function can be used to estimate genetic and residual covariance and correlations for DS, Cholesky, IP and IPC models, based on grmseml.fit or grmseml.stpar objects. For the latter, the diagonal elements of the VA output matrix detail the heritabilities. Except for directly estimated variance components using the DS model, all standard errors are derived with the Delta method.

Value

grmseml.var returns a list object consisting of the following matrices:

- VA estimated genetic variance
- VA.se standard error of estimated genetic variance
- VE estimated residual variance
- VE.se standard error of estimated residual variance
- VP estimated total phenotypic variance
- RG genetic correlation
- RG.se standard error genetic correlation
- RE residual correlation
- RG.se standard error residual correlation

Examples

```
 #(runtime should be less than one minute)

 out <- grmsem.fit(ph.small, G.small, LogL = TRUE, estSE = TRUE)
 var.out <- grmsem.var(out)
 print(var.out)
```

ph.large *Phenotype data: large data set*

Description

A large quad-variate data set was simulated assuming an underlying Cholesky model, with 5000 observations per trait and high polygenicity (5,000 SNPs per genetic factor). The data set is described in full in the vignette. Genetic and residual covariances are assumed to be influenced by four independent genetic factors (A1, A2, A3 and A4, based on 5000 SNPs each) and four independent residual factors (E1, E2, E3 and E4), respectively.

Usage

```
ph.large
```

Format

A data frame with 5000 rows and 4 columns

Y1 trait one

Y2 trait two

Y3 trait three

Y4 trait four

ph.small *Phenotype data: small data set*

Description

A small tri-variate data set was simulated assuming an underlying Cholesky model, with 100 observations per trait and low polygenicity (150 SNPs per genetic factor). The data set is described in full in the vignette. Genetic and residual covariances are assumed to be influenced by three independent genetic factors (A1, A2 and A3, based on a 150 SNPs each) and three independent residual factors (E1, E2 and E3), respectively.

Usage

```
ph.small
```

Format

A data frame with 100 rows and 3 columns

Y1 trait one

Y2 trait two

Y3 trait three

Index

* datasets

fit.large, 2

G.small, 3

ph.large, 12

ph.small, 12

* grmsem

grm.input, 4

grmsem.biher, 4

grmsem.fcoher, 5

grmsem.fit, 6

grmsem.stpar, 10

grmsem.var, 11

fit.large, 2

G.small, 3

grm.bin.input, 3

grm.input, 4

grmsem.biher, 4

grmsem.fcoher, 5

grmsem.fit, 6

grmsem.stpar, 10

grmsem.var, 11

ph.large, 12

ph.small, 12

readBin, 3