User Manual for GMA

Chao Ning

China Agricultural University

ningchao@cau.edu.cn; ningchao91@gmail.com

Introduction

GMA is a series of efficient multivariate analysis algorithms for genome-wide association studies (GWAS) and genomic selection (GS). Different analysis algorithms will be constantly updated. We write GMA with Python now and consider rewriting it with C++ in the future.

Installation

GMA depends on a series of packages. We recommend installing Anaconda (Python 2.7 version; https://www.anaconda.com/download/) to get all the required package dependencies. Please run command of pip install package_name to install nonexistent packages.

Then, go to the directory of GMA and type python setup.py install to install GMA.

GMA functions

New functions will be constantly included.

1. Longitudinal Data Analysis

1.1 gma.cal_kin

gma.cal_kin(bed_file, inv=True, small_val=0.001)

Function: Build the genomic additive relationship matrix (VanRaden, 2008) and its inversion (optional) using plink binary ped file (*.fam, *.bim, and *.bed). Note that **no missing genotypes** are allowed in current version. The software BEAGLE or IMPUTE2 can be used to fill the missing genotype. When the accuracy of the filled-in calls isn't important, the PLINK command --fill-missing-a2 can be used to simply replace all missing calls with homozygous A2 calls, which may have little influence for relative low missing rate (eg. Less than 0.05 for each SNP).

Parameters:

bed_file: the prefix of plink binary ped file;

inv: whether to calculate the inversion of genomic additive relationship matrix, default value is True;

small_val: small value that is added to the diagonal to guarantee the positive matrix, default value is 0.001.

Returns:

A List: contain two square matrixes. The first one is the genomic additive relationship matrix and the second one is the inversion (inv=True) or None (inv=False).

File(s): The program will generate one or two file(s) of the same prefix with plink binary ped file: genomic additive relationship matrix (*.grm) and its inversion (*.giv; while inv=True). The file contains the lower triangle elements of matrix, including three columns of ID, ID, and value.

Example:

```
>>>from gma import cal_kin
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
```

1.2 gma.balance_varcom

gma.balance_varcom(data_file, id, tpoint, trait, kin_file, tfix=None, fix=None, forder=3, rorder=3, na_method='omit', init=None, maxiter=200, cc_par=1.0e-8, cc_gra=1.0e6, cc_ll=1.0e6, em_weight_step=0.01, prefix_outfile='gma_balance_varcom')

Function: Estimate the variance components for random regression model with eigen-decomposition technique. The program is applicable to balanced longitudinal data (every individual is recorded at the same time points).

Parameters:

data_file: a string. The file name of data file. The data file must include header line, which indicates variate names. The variates whose names begin with a capital letter in the header line will be automatically converted into factors (classified variable), while the variates whose names begin with a lowercase letter in the header line will be automatically converted into covariates (continuous variable).

id: a string. The variate name existing in header line indicates the individual column. It must begin with a capital letter.

tpoint: a list. The time points recording the phenotypic values.

trait: a list. The column numbers for the phenotypic values. The column number starts from 0 in the data file.

kin_file: a string. The file name of genomic relationship matrix.

tfix: a string. In order to improve computational efficiency, only default value of None is allowed in current version.

fix: a string. In order to improve computational efficiency, only default value of None is allowed in current version.

forder: an integer. Default value is 3. The order of legendre polynomials for fix regression (time-varied mean).

rorder: An integer. Default value is 3. The order of legendre polynomials for random regression (including additive genetic effect and permanent environment effect).

na_method: A sting, 'omit' or 'include'. Default value is 'omit'. It indicates how to deal with the missing values (NaN or NA). The value of 'omit' means that the program will remove the row if any variate is missing. The value of 'include' means that the program will fill the missing value with previous value of the variate.

init: A list. Default value is None. The initial values for variance components. It includes only the lower triangle elements of covariance structure.

maxiter: An integer. Default value is 200. The maximum iteration numbers for variance components estimation.

cc_par: float. Default value is 1.0e-8. The convergence criteria for change in parameters.

cc_gra: float. Default value is 1.0e6. The convergence criteria for norm of gradient vector (first partial derivatives). cc_gra does not work in default value.

cc_ll: float. Default value is 1.0e6. The convergence criteria for change in -2*logL.

cc_ll does not work in default value.

em_weight_step: float. Default value is 0.01. The em information matrix weight moving from 0 to 1 with defined step.

prefix_outfile: A string. Default value is 'gma_balance_varcom'. The prefix for output file.

Returns:

A dictionary: including a pandas data frame of variance components ('variances'), change in parameters in the last iteration ('cc_par'), norm of gradient vector ('cc_gra'), -2*logL value ('ll'), whether converge ('convergence'), AIC value ('AIC') and BIC value ('BIC').

File(s): The log file (*.log) and the variance components file (*.var).

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import balance_varcom
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.balance.txt'
>>>id = 'ID'
>>>tpoint = np.array(range(16)) + 1.0
>>>trait = range(2, 18)
>>>kin_file = './tests/plink.grm'
>>>prefix_outfile = './tests/gma_balance_varcom'
>>>res_var = balance_varcom(data_file, id, tpoint, trait, kin_file,
prefix_outfile=prefix_outfile)
>>>print res_var
# If the variances do not converge, we can use the previous variances as initial values.
>>>init = np.array(res_var['variances']['var_val'])
# or
# var_com = pd.read_csv('./tests/gma_balance_varcom.var', header=0, sep='\s+')
# init = np.array(var_com['var_val'])
>>>res_var = balance_varcom(data_file, id, tpoint, trait, kin_file, init=init,
prefix_outfile=prefix_outfile)
>>>print res_var
```

1.3 gma.balance_longwas_fixreg

gma.balance_longwas_fixreg(data_file, id, tpoint, trait, kin_file, bed_file, var_com, snp_lst=None, tfix=None, fix=None, forder=3, rorder=3, na_method='omit', maxiter=10, cc_par=1.0e-6, cc_gra=1000000.0, em_weight_step=0.01, prefix_outfile='gma_balance_longwas_fixreg')

Function: Perform longitudinal GWAS for balanced data. The program fit SNP effect as fix regression (time-varied) and will optimize the variance components per-SNP. As most of SNPs contribute little to the phenotypic values, the program take the variance component in the null model (without SNP effect, from gma.balance_varcom program) as initial values. Thus, relevant parameters should be same for gma.balance_varcom and gma.balance_longwas_fixreg program.

Parameters:

data_file: a string. The file name of data file. The data file must include header line, which indicates variate names. The variates whose names begin with a capital letter in the header line will be automatically converted into factors (classified variable), while the variates whose names begin with a lowercase letter in the header line will be automatically converted into covariates (continuous variable).

id: a string. The variate name existing in header line indicates the individual column. It must begin with a capital letter.

tpoint: a list. The time points recording the phenotypic values.

trait: a list. The column numbers for the phenotypic values. The column number starts from 0 in the data file.

kin_file: a string. The file name of genomic relationship matrix.

bed_file: a string. the prefix of plink binary ped file. No missing genotype is allowed.

var_com: a pandas data frame of variance components from gma.balance_varcom program.

snp_lst: list. Default value is None. The SNP order list to test. None means all the SNP will be tested. The elements in the list is more or equal to 0 and less than then number of SNPs.

tfix: a string. In order to improve computational efficiency, only default value of None is allowed in current version.

fix: a string. In order to improve computational efficiency, only default value of None is allowed in current version.

forder: an integer. Default value is 3. The order of legendre polynomials for fix regression (time-varied mean).

rorder: An integer. Default value is 3. The order of legendre polynomials for random regression (including additive genetic effect and permanent environment effect).

na_method: A sting, 'omit' or 'include'. Default value is 'omit'. It indicates how to deal with the missing values (NaN or NA). The value of 'omit' means that the program will remove the row if any variate is missing. The value of 'include' means that the program will fill the missing value with previous value of the variate.

maxiter: An integer. Default value is 200. The maximum iteration numbers for variance components estimation.

cc_par: float. Default value is 1.0e-6. The convergence criteria for change in parameters.

cc_gra: float. Default value is 1.0e6. The convergence criteria for norm of gradient vector (first partial derivatives). cc_gra does not work in default value.

em_weight_step: float. Default value is 0.01. The em information matrix weight moving from 0 to 1 with defined step.

prefix_outfile: A string. Default value is ' gma_balance_longwas_fixreg'. The prefix for output file.

Returns:

Pandas data frame: one row per test SNP.

File(s): The log file (*.log) and the test SNP result file (*.res).

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import balance_varcom
>>>from gma import balance_longwas_fixreg
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.balance.txt'
>>>id = 'ID'
>>>tpoint = np.array(range(16)) + 1.0
>>>trait = range(2, 18)
>>>kin_file = './tests/plink.grm'
>>>prefix_outfile = './tests/gma_balance_varcom'
>>>res_var = balance_varcom(data_file, id, tpoint, trait, kin_file,
prefix_outfile=prefix_outfile)
>>>var_com = res_var['variances']
# or
# var_com = pd.read_csv('./tests/gma_balance_varcom.var', header=0, sep='\s+')
>>>snp_lst = range(1, 10)
# snp_lst = [2, 3, 5, 10]
>>>prefix_outfile = './tests/gma_balance_longwas_fixreg'
>>>res_lst = balance_longwas_fixreg(data_file, id, tpoint, trait, kin_file, bed_file,
var_com, snp_lst=snp_lst, prefix_outfile=prefix_outfile)
```

1.4 gma.balance_longwas_lt

gma.balance_longwas_lt(data_file, id, tpoint, trait, kin_file, bed_file, var_com, snp_lst=None, tfix=None, fix=None, forder=3, rorder=3, na_method='omit', prefix_outfile='gma_balance_longwas_lt')

Function: Perform longitudinal GWAS for balanced data. The program obtain the time-varied SNP effect by linear transformation of genomic estimated values algorithm. Fast but less powerful. The program take advantage of the variance component from gma.balance_varcom program. Thus, relevant parameters should be same for gma.balance_varcom and gma.balance_longwas_lt program.

Parameters:

data_file: a string. The file name of data file. The data file must include header line, which indicates variate names. The variates whose names begin with a capital letter in the header line will be automatically converted into factors (classified variable), while the variates whose names begin with a lowercase letter in the header line will be automatically converted into covariates (continuous variable).

id: a string. The variate name existing in header line indicates the individual column. It must begin with a capital letter.

tpoint: a list. The time points recording the phenotypic values.

trait: a list. The column numbers for the phenotypic values. The column number starts from 0 in the data file.

kin_file: a string. The file name of genomic relationship matrix.

bed_file: a string. the prefix of plink binary ped file. No missing genotype is allowed.

var_com: a pandas data frame of variance components from gma.balance_varcom program.

snp_lst: list. Default value is None. The SNP order list to test. None means all the SNP will be tested. The elements in the list is more or equal to 0 and less than then number of SNPs.

tfix: a string. In order to improve computational efficiency, only default value of None is allowed in current version.

fix: a string. In order to improve computational efficiency, only default value of None is allowed in current version.

forder: an integer. Default value is 3. The order of legendre polynomials for fix regression (time-varied mean).

rorder: An integer. Default value is 3. The order of legendre polynomials for random regression (including additive genetic effect and permanent environment effect).

na_method: A sting, 'omit' or 'include'. Default value is 'omit'. It indicates how to deal with the missing values (NaN or NA). The value of 'omit' means that the program will remove the row if any variate is missing. The value of 'include' means that the program will fill the missing value with previous value of the variate.

prefix_outfile: A string. Default value is 'gma_balance_longwas_lt '. The prefix for output file.

Returns:

Pandas data frame: one row per test SNP.

File(s): The log file (*.log) and the test SNP result file (*.res).

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import balance_varcom
>>>from gma import balance_longwas_lt
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.balance.txt'
>>>id = 'ID'
>>>tpoint = np.array(range(16)) + 1.0
>>>trait = range(2, 18)
>>>kin_file = './tests/plink.grm'
>>>prefix_outfile = './tests/gma_balance_varcom'
>>>res_var = balance_varcom(data_file, id, tpoint, trait, kin_file,
prefix_outfile=prefix_outfile)
>>>var_com = res_var['variances']
# or
# var_com = pd.read_csv('./tests/gma_balance_varcom.var', header=0, sep='\s+')
>>>snp_lst = range(1, 10)
```

```
# snp_lst = [2, 3, 5, 10]
>>>prefix_outfile = './tests/gma_balance_longwas_lt'
>>>res_lst = balance_longwas_lt(data_file, id, tpoint, trait, kin_file, bed_file,
var_com, snp_lst=snp_lst, prefix_outfile=prefix_outfile)
```

1.5 gma.balance_longwas_lt + gma.balance_longwas_fixreg

Function: Perform longitudinal GWAS for balanced data in an rapid and powerful way. Perform gma.balance_longwas_lt firstly, and then select SNPs less than a certain *P*-values (such as 0.01) to calculate their gma.balance_longwas_fixreg *P*-values.

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import balance_varcom
>>>from gma import balance_longwas_lt
>>>from gma import balance_longwas_fixreg
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.balance.txt'
>>>id = 'ID'
>>>tpoint = np.array(range(16)) + 1.0
>>>trait = range(2, 18)
>>>kin_file = './tests/plink.grm'
>>>prefix_outfile = './tests/gma_balance_varcom'
>>>res_var = balance_varcom(data_file, id, tpoint, trait, kin_file,
prefix_outfile=prefix_outfile)
>>>var_com = res_var['variances']
>>># or
>>># var_com = pd.read_csv('./tests/gma_balance_varcom.var', header=0, sep='\s+')
```

```
>>>prefix_outfile = './tests/gma_balance_longwas_lt'
>>>res_lst = balance_longwas_lt(data_file, id, tpoint, trait, kin_file, bed_file,
var_com, snp_lst=None, prefix_outfile=prefix_outfile)
>>>res_lst_sel = res_lst[res_lst['p_val'] < 0.01]
>>>snp_lst = np.array(res_lst_sel['order'])
>>>prefix_outfile = './tests/gma_balance_longwas_fixreg_sel'
>>>res_lst_fixreg_sel = balance_longwas_fixreg(data_file, id, tpoint, trait, kin_file,
bed_file, var_com, snp_lst=snp_lst, prefix_outfile=prefix_outfile)
```

1.6 gma.unbalance_varcom

gma.unbalance_varcom(data_file, id, tpoint, trait, kin_inv_file, tfix=None, fix=None, forder=3, aorder=3, porder=3, na_method='omit', fixcon=False, rancon=True, init=None, maxiter=200, cc_par=1.0e-8, cc_gra=1.0e6, cc_ll=1.0e6, em_weight_step=0.01, prefix_outfile='gma_unbalance_varcom')

Function: Estimate the variance components for random regression model. The program is applicable to unbalanced longitudinal data (every individual can be recorded at the different time points).

Parameters:

data_file: a string. The file name of data file. The data file must include header line, which indicates variate names. The variates whose names begin with a capital letter in the header line will be automatically converted into factors (classified variable), while the variates whose names begin with a lowercase letter in the header line will be automatically converted into covariates (continuous variable).

id: a string. The variate name existing in header line indicates the individual column. It must begin with a capital letter. Different from balanced longitudinal data file in which each individual has one column, repeat columns exist (the number is same to the recorded measures). The data file must sorted by tpoint within id.

tpoint: a string. The variate name existing in header line indicates the time points column.

trait: a string. The variate name existing in header line indicates the phenotypic values column.

kin_inv_file: a string. The file name for the inversion of genomic relationship matrix.

tfix: a string. Default value is None. The variate name existing in header line indicates time-dependent fix effect. It must begin with a capital letter. This means different level has different time-varied means.

fix: a string. Default values is None. The program use the patsy.dmatrix function to build the design matrix. For factors (variates whose names begin with a capital letter), C() function must be used. Here is an example, fix='C(Sex)+age'.

forder: an integer. Default value is 3. The order of legendre polynomials for fix regression (time-varied mean).

aorder: An integer. Default value is 3. The order of legendre polynomials for additive genetic effect.

porder: An integer. Default value is 3. The order of legendre polynomials for permanent environment effect.

na_method: A sting, 'omit' or 'include'. Default value is 'omit'. It indicates how to deal with the missing values (NaN or NA). The value of 'omit' means that the program will remove the row if any variate is missing. The value of 'include' means that the program will fill the missing value with previous value of the variate.

fixcon: Boolean. Default value is False. Whether include the fix constant term in the log likelihood value.

rancon: Boolean. Default value is True. Whether include the random constant term in the log likelihood value.

init: A list. Default value is None. The initial values for variance components. It includes only the lower triangle elements of covariance structure.

maxiter: An integer. Default value is 200. The maximum iteration numbers for variance components estimation.

cc_par: float. Default value is 1.0e-8. The convergence criteria for change in parameters.

cc_gra: float. Default value is 1.0e6. The convergence criteria for norm of gradient vector (first partial derivatives). cc_gra does not work in default value.

cc_ll: float. Default value is 1.0e6. The convergence criteria for change in -2*logL. cc_ll does not work in default value.

em_weight_step: float. Default value is 0.01. The em information matrix weight moving from 0 to 1 with defined step.

prefix_outfile: A string. Default value is 'gma_unbalance_varcom'. The prefix for output file.

Returns:

A dictionary: including a pandas data frame of variance components ('variances'), change in parameters in the last iteration ('cc_par'), norm of gradient vector ('cc_gra'), -2*logL value ('ll'), whether converge ('convergence'), the effect vector ('effect'), the inversion of coefficient matrix ('CMi'), AIC value ('AIC') and BIC value ('BIC').

File(s): The log file (*.log) and the variance components file (*.var).

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import unbalance_varcom
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.unbalance.txt'
>>>id = 'ID'
>>>tpoint = 'weak'
>>>trait = 'trait'
>>>kin_inv_file = './tests/plink.giv'
>>>tfix = 'Sex'
>>>prefix_outfile = './tests/gma_unbalance_varcom'
>>>res_var = unbalance_varcom(data_file, id, tpoint, trait, kin_inv_file, tfix=tfix,
prefix_outfile=prefix_outfile)
>>>print res_var
```

1.7 gma.unbalance_longwas_fixreg

gma.unbalance_longwas_fixreg(data_file, id, tpoint, trait, bed_file, kin_file, kin_inv_file, var_com, snp_lst=None, tfix=None, fix=None, forder=3, aorder=3, porder=3, na_method='omit', prefix_outfile='gma_unbalance_longwas_fixreg')

Function: Perform longitudinal GWAS for unbalanced data. The program fit SNP effect as fix regression (time-varied) and use the population parameters previously determined (P3D) algorithm. As most of SNPs contribute little to the phenotypic values, the program utilize the variance components in the null model (without SNP effect, from gma.unbalance_varcom program). Thus, relevant parameters should be same for gma.unbalance_varcom and gma.unbalance_longwas_fixreg program.

Parameters:

data_file: a string. The file name of data file. The data file must include header line, which indicates variate names. The variates whose names begin with a capital letter in the header line will be automatically converted into factors (classified variable), while the variates whose names begin with a lowercase letter in the header line will be automatically converted into covariates (continuous variable).

id: a string. The variate name existing in header line indicates the individual column. It must begin with a capital letter. Different from balanced longitudinal data file in which each individual has one column, repeat columns exist (the number is same to the recorded measures). The data file must sorted by tpoint within id.

tpoint: a string. The variate name existing in header line indicates the time points column.

trait: a string. The variate name existing in header line indicates the phenotypic values column.

bed_file: a string. the prefix of plink binary ped file. No missing genotype is allowed.

kin_file: a string. The file name for the genomic relationship matrix.

kin_inv_file: a string. The file name for the inversion of genomic relationship matrix.

var_com: a pandas data frame of variance components from gma.balance_varcom program.

snp_lst: list. Default value is None. The SNP order list to test. None means all the SNP will be tested. The elements in the list is more or equal to 0 and less than then number of SNPs.

tfix: a string. Default value is None. The variate name existing in header line indicates time-dependent fix effect. It must begin with a capital letter. This means different level has different time-varied means.

fix: a string. Default values is None. The program use the patsy.dmatrix function to build the design matrix. For factors (variates whose names begin with a capital letter), C() function must be used. Here is an example, fix='C(Sex)+age'.

forder: an integer. Default value is 3. The order of legendre polynomials for fix regression (time-varied mean).

aorder: An integer. Default value is 3. The order of legendre polynomials for additive genetic effect.

porder: An integer. Default value is 3. The order of legendre polynomials for permanent environment effect.

na_method: A sting, 'omit' or 'include'. Default value is 'omit'. It indicates how to deal with the missing values (NaN or NA). The value of 'omit' means that the program will remove the row if any variate is missing. The value of 'include' means that the program will fill the missing value with previous value of the variate.

prefix_outfile: A string. Default value is 'gma_unbalance_longwas_fixreg'. The prefix for output file.

Returns:

Pandas data frame: one row per test SNP.

File(s): The log file (*.log) and the test SNP result file (*.res).

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import unbalance_varcom
>>>from gma import unbalance_longwas_fixreg
```

```
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.unbalance.txt'
>>>id = 'ID'
>>>tpoint = 'weak'
>>>trait = 'trait'
>>>kin_inv_file = './tests/plink.giv'
>>>tfix = 'Sex'
>>>prefix_outfile = './tests/gma_unbalance_varcom'
>>>res_var = unbalance_varcom(data_file, id, tpoint, trait, kin_inv_file, tfix=tfix,
prefix_outfile=prefix_outfile)
>>>print res_var
>>>kin_file = './tests/plink.grm'
>>>var_com = pd.read_csv("./tests/gma_unbalance_varcom.var", sep='\s+',
header=0)
>>>prefix_outfile = './tests/gma_unbalance_longwas_fixreg'
>>>res_lst = unbalance_longwas_fixreg(data_file, id, tpoint, trait, bed_file, kin_file,
kin_inv_file, var_com, tfix=tfix, prefix_outfile='gma_unbalance_longwas_fixreg')
```

1.8 gma.unbalance_longwas_lt

gma.unbalance_longwas_lt(data_file, id, tpoint, trait, bed_file, kin_file, kin_inv_file, var_com, snp_lst=None, tfix=None, fix=None, forder=3, aorder=3, porder=3, na_method='omit', prefix_outfile='gma_unbalance_longwas_lt')

Function: Perform longitudinal GWAS for unbalanced data. The program obtain the time-varied SNP effect by linear transformation of genomic estimated values algorithm. The program take advantage of the variance component from gma.unbalance_varcom program. Thus, relevant parameters should be same for gma.unbalance_varcom and gma.unbalance_longwas_lt program.

.

Parameters:

data_file: a string. The file name of data file. The data file must include header line, which indicates variate names. The variates whose names begin with a capital letter in the header line will be automatically converted into factors (classified variable), while the variates whose names begin with a lowercase letter in the header line will be automatically converted into covariates (continuous variable).

id: a string. The variate name existing in header line indicates the individual column. It must begin with a capital letter. Different from balanced longitudinal data file in which each individual has one column, repeat columns exist (the number is same to the recorded measures). The data file must sorted by tpoint within id.

tpoint: a string. The variate name existing in header line indicates the time points column.

trait: a string. The variate name existing in header line indicates the phenotypic values column.

bed_file: a string. the prefix of plink binary ped file. No missing genotype is allowed.

kin_file: a string. The file name for the genomic relationship matrix.

kin_inv_file: a string. The file name for the inversion of genomic relationship matrix.

var_com: a pandas data frame of variance components from gma.balance_varcom program.

snp_lst: list. Default value is None. The SNP order list to test. None means all the SNP will be tested. The elements in the list is more or equal to 0 and less than then number of SNPs.

tfix: a string. Default value is None. The variate name existing in header line indicates time-dependent fix effect. It must begin with a capital letter. This means different level has different time-varied means.

fix: a string. Default values is None. The program use the patsy.dmatrix function to build the design matrix. For factors (variates whose names begin with a capital letter), C() function must be used. Here is an example, fix='C(Sex)+age'.

forder: an integer. Default value is 3. The order of legendre polynomials for fix regression (time-varied mean).

aorder: An integer. Default value is 3. The order of legendre polynomials for additive genetic effect.

porder: An integer. Default value is 3. The order of legendre polynomials for permanent environment effect.

na_method: A sting, 'omit' or 'include'. Default value is 'omit'. It indicates how to deal with the missing values (NaN or NA). The value of 'omit' means that the program will remove the row if any variate is missing. The value of 'include' means that the program will fill the missing value with previous value of the variate.

prefix_outfile: A string. Default value is 'gma_unbalance_longwas_lt'. The prefix for output file.

Returns:

Pandas data frame: one row per test SNP.

File(s): The log file (*.log) and the test SNP result file (*.res).

Example:

```
>>>import numpy as np
>>>import pandas as pd
>>>from gma import cal_kin
>>>from gma import unbalance_varcom
>>>from gma import unbalance_longwas_lt
>>>bed_file = './tests/plink'
>>>kin_lst = cal_kin(bed_file)
>>>data_file = './tests/phe.unbalance.txt'
>>>id = 'ID'
>>>tpoint = 'weak'
>>>trait = 'trait'
>>>kin_inv_file = './tests/plink.giv'
>>>tfix = 'Sex'
>>>prefix_outfile = './tests/gma_unbalance_varcom'
>>>res_var = unbalance_varcom(data_file, id, tpoint, trait, kin_inv_file, tfix=tfix,
prefix_outfile=prefix_outfile)
```

```
>>>print res_var
>>>kin_file = './tests/plink.grm'
>>>var_com = pd.read_csv("./tests/gma_unbalance_varcom.var", sep='\s+',
header=0)
>>>prefix_outfile = './tests/gma_unbalance_longwas_lt'
>>>res_lst = unbalance_longwas_lt(data_file, id, tpoint, trait, bed_file, kin_file,
kin_inv_file, var_com, tfix=tfix, prefix_outfile=prefix_outfile)
```

VanRaden, P.M. (2008) Efficient methods to compute genomic predictions, *Journal of dairy science*, **91**, 4414-4423.