# Package 'rMVP'

January 10, 2025

**Type** Package

**Title** Memory-Efficient, Visualize-Enhanced, Parallel-Accelerated GWAS Tool

**Version** 1.3.5

**Date** 2024-12-30

**Description** A memory-efficient, visualize-enhanced, parallel-accelerated Genome-Wide Association Study (GWAS) tool. It can
(1) effectively process large data,
(2) rapidly evaluate population structure,
(3) efficiently estimate variance components several algorithms,
(4) implement parallel-accelerated association tests of markers three methods,
(5) globally efficient design on GWAS process computing,
(6) enhance visualization of related information.
'rMVP' contains three models GLM (Alkes Price (2006) <DOI:10.1038/ng1847>), MLM (Jianming Yu (2006) <DOI:10.1038/ng1702>)
and FarmCPU (Xiaolei Liu (2016) <doi:10.1371/journal.pgen.1005767>); variance components estimation methods EMMAX
(Hyunmin Kang (2008) <DOI:10.1534/genetics.107.080101>;), FaSTLMM (method: Christoph Lippert (2011) <DOI:10.1038/nmeth.1681>,
R implementation from 'GAPIT2': You Tang and Xiaolei Liu (2016) <DOI:10.1371/journal.pone.0107684> and
'SUPER': Qishan Wang and Feng Tian (2014) <DOI:10.1371/journal.pone.0107684>), and HE regression (Xiang Zhou (2017) <DOI:10.1214/17-AOAS1052>).

**License** Apache License 2.0

**Encoding** UTF-8

**URL** https://github.com/xiaolei-lab/rMVP

**BugReports** https://github.com/xiaolei-lab/rMVP/issues

**Imports** utils, stats, methods, graphics, grDevices, MASS, bigmemory, RhpcBLASctl

**Depends** R (>= 3.3)

**LinkingTo** Rcpp, RcppArmadillo, RcppEigen, RcppProgress, BH, bigmemory

**NeedsCompilation** yes

**Suggests** knitr, testthat, rmarkdown

**RoxygenNote** 7.3.2

**Maintainer** Xiaolei Liu <xll198708@gmail.com>

**Author** Lilin Yin [aut],
Haohao Zhang [aut],
Zhenshuang Tang [aut],
Jingya Xu [aut],
Dong Yin [aut],
Zhiwu Zhang [aut],
Xiaohui Yuan [aut],
Mengjin Zhu [aut],
Shuhong Zhao [aut],
Xinyun Li [aut],
Qishan Wang [ctb],
Feng Tian [ctb],
Hyunmin Kang [ctb],
Xiang Zhou [ctb],
Xiaolei Liu [cre, aut, cph]

**Repository** CRAN

**Date/Publication** 2025-01-10 04:10:02 UTC

# Contents

---

MVP *MVP, Memory-efficient, Visualization-enhanced, Parallel-accelerated*

---

## Description

Object 1: To perform GWAS using General Linear Model (GLM), Mixed Linear Model (MLM), and FarmCPU model Object 2: To calculate kinship among individuals using Varaden method Object 3: Estimate variance components using EMMA, FaST-LMM, and HE regression Object 4: Generate high-quality figures

## Usage

```
MVP(
  phe,
  geno,
  map,
  K = NULL,
  nPC.GLM = NULL,
  nPC.MLM = NULL,
  nPC.FarmCPU = NULL,
  CV.GLM = NULL,
  CV.MLM = NULL,
  CV.FarmCPU = NULL,
  REML = NULL,
  maxLine = 10000,
  ncpus = detectCores(logical = FALSE),
  vc.method = c("BRENT", "EMMA", "HE"),
  method = c("GLM", "MLM", "FarmCPU"),
  maf = NULL,
  p.threshold = NA,
  QTN.threshold = 0.01,
  method.bin = "static",
  bin.size = c(5e+05, 5e+06, 5e+07),
  bin.selection = seq(10, 100, 10),
  maxLoop = 10,
  permutation.threshold = FALSE,
  permutation.rep = 100,
  memo = NULL,
  outpath = getwd(),
```

```
 col = c("#4197d8", "#f8c120", "#413496", "#495226", "#d60b6f", "#e66519", "#d581b7",
    "#83d3ad", "#7c162c", "#26755d"),
  file.output = TRUE,
  file.type = "jpg",
  dpi = 300,
  threshold = 0.05,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| phe | phenotype, n * 2 matrix, n is sample size |
| geno | genotype, either m by n or n by m is supportable, m is marker size, n is population size |
| map | SNP map information, SNP name, Chr, Pos |
| K | Kinship, Covariance matrix(n * n) for random effects, must be positive semi-definite |
| nPC.GLM | number of PCs added as fixed effects in GLM |
| nPC.MLM | number of PCs added as fixed effects in MLM |
| nPC.FarmCPU | number of PCs added as fixed effects in FarmCPU |
| CV.GLM | covariates added in GLM |
| CV.MLM | covariates added in MLM |
| CV.FarmCPU | covariates added in FarmCPU |
| REML | a list contains ve and vg |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| ncpus | number of cpus used for parallel |
| vc.method | methods for estimating variance component("EMMA" or "HE" or "BRENT") |
| method | the GWAS model, "GLM", "MLM", and "FarmCPU", models can be selected simutaneously, i.e. c("GLM", "MLM", "FarmCPU") |
| maf | the threshold of minor allele frequency to filter SNPs in analysis |
| p.threshold | if all p values generated in the first iteration are bigger than p.threshold, FarmCPU stops |
| QTN.threshold | in second and later iterations, only SNPs with lower p-values than QTN.threshold have chances to be selected as pseudo QTNs |
| method.bin | 'static' or 'FaST-LMM' |
| bin.size | window size in genome |
| bin.selection | a vector, how many windows selected |
| maxLoop | maximum number of iterations |
| permutation.threshold | if use a permutation cutoff or not (bonferroni cutoff) |

| | |
|---|---|
| permutation.rep | number of permutation replicates |
| memo | Character. A text marker on output files |
| outpath | the path of the output files |
| col | for color of points in each chromosome on manhattan plot |
| file.output | whether to output files or not |
| file.type | figure formats, "jpg", "tiff" |
| dpi | resolution for output figures |
| threshold | a cutoff line on manhattan plot, 0.05/marker size |
| verbose | whether to print detail. |

## Value

Output: MVP.return$map - SNP map information, SNP name, Chr, Pos Output: MVP.return$glm.results - p-values obtained by GLM method Output: MVP.return$mlm.results - p-values obtained by MLM method Output: MVP.return$farmcpu.results - p-values obtained by FarmCPU method

## Author(s)

Lilin Yin, Haohao Zhang, and Xiaolei Liu

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
print(dim(genotype))
mapPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.map", package = "rMVP")
map <- read.table(mapPath , head = TRUE)

opts <- options(rMVP.OutputLog2File = FALSE)

mvp <- MVP(phe=phenotype, geno=genotype, map=map, maxLoop=3,
  method=c("GLM", "MLM", "FarmCPU"), file.output=FALSE, ncpus=1)
str(mvp)

options(opts)
```

`MVP.BRENT.Vg.Ve` *MVP.BRENT.Vg.Ve variance component estimation using the BRENT method*

## Description

MVP.BRENT.Vg.Ve variance component estimation using the BRENT method

## Usage

```
MVP.BRENT.Vg.Ve(y, X, eigenK, verbose = FALSE)
```

## Arguments

| | |
|---|---|
| y | phenotype |
| X | covariate matrix, the first column is 1s |
| eigenK | eigen of Kinship matrix |
| verbose | whether to print detail. |

## Value

vg, ve, and delta

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
print(dim(genotype))

eigenK <- eigen(MVP.K.VanRaden(genotype, cpu=1))
vc <- MVP.BRENT.Vg.Ve(y=phenotype[,2], X=matrix(1, nrow(phenotype)), eigenK=eigenK)
print(vc)
```

MVP.Data *MVP.Data: To prepare data for MVP package*

## Description

MVP.Data: To prepare data for MVP package

## Usage

```
MVP.Data(
  fileMVP = NULL,
  fileVCF = NULL,
  fileHMP = NULL,
  fileBed = NULL,
  fileNum = NULL,
  fileMap = NULL,
  filePhe = NULL,
  fileInd = NULL,
  fileKin = NULL,
  filePC = NULL,
  out = "mvp",
  sep.num = "\t",
  auto_transpose = TRUE,
  sep.map = "\t",
  sep.phe = "\t",
  sep.kin = "\t",
  sep.pc = "\t",
  type.geno = "char",
  pheno_cols = NULL,
  SNP.impute = "Major",
  maxLine = 10000,
  pcs.keep = 5,
  verbose = TRUE,
  ncpus = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| fileMVP | Genotype in MVP format |
| fileVCF | Genotype in VCF format |
| fileHMP | Genotype in hapmap format |
| fileBed | Genotype in PLINK binary format |
| fileNum | Genotype in numeric format; pure 0, 1, 2 matrix; m * n or n * m, m is marker size, n is sample size |

| | |
|---|---|
| fileMap | SNP map information, there are three columns, including SNP_ID, Chromosome, and Position |
| filePhe | Phenotype, the first column is taxa name, the subsequent columns are traits |
| fileInd | Individual name file |
| fileKin | Kinship that represents relationship among individuals, n * n matrix, n is sample size |
| filePC | Principal components, n*npc, n is sample size, npc is number of top columns of principal components |
| out | prefix of output file name |
| sep.num | seperator for numeric file. |
| auto_transpose | Whether to automatically transpose numeric genotypes, the default is True, which will identify the most one of the rows or columns as a marker, If set to False, the row represents the marker and the column represents the individual. |
| sep.map | seperator for map file. |
| sep.phe | seperator for phenotype file. |
| sep.kin | seperator for Kinship file. |
| sep.pc | seperator for PC file. |
| type.geno | type parameter in bigmemory, genotype data. The default is char, it is highly recommended *NOT* to modify this parameter. |
| pheno_cols | Extract which columns of the phenotype file (including individual IDs) |
| SNP.impute | "Left", "Middle", "Right", or NULL for skip impute. |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| pcs.keep | how many PCs to keep |
| verbose | whether to print detail. |
| ncpus | The number of threads used, if NULL, (logical core number - 1) is automatically used |
| ... | Compatible with DEPRECATED parameters. |

## Value

NULL Output files: genotype.desc, genotype.bin: genotype file in bigmemory format phenotype.phe: ordered phenotype file, same taxa order with genotype file map.map: SNP information k.desc, k.bin: Kinship matrix in bigmemory format pc.desc, pc.bin: PC matrix in bigmemory format Requirement: fileHMP, fileBed, and fileNum can not input at the same time

## Examples

```
bfilePath <- file.path(system.file("extdata", "02_bfile", package = "rMVP"), "mvp")
opts <- options(rMVP.OutputLog2File = FALSE)

MVP.Data(fileBed=bfilePath, out=tempfile("outfile"), ncpus=1)

options(opts)
```

`MVP.Data.Bfile2MVP` *MVP.Data.Bfile2MVP: To transform plink binary data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018*

## Description

MVP.Data.Bfile2MVP: To transform plink binary data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.Bfile2MVP(
  bfile,
  out = "mvp",
  maxLine = 10000,
  type.geno = "char",
  threads = 0,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `bfile` | Genotype in binary format (.bed, .bim, .fam) |
| `out` | the name of output file |
| `maxLine` | the number of markers handled at a time, smaller value would reduce the memory cost |
| `type.geno` | the type of genotype elements |
| `threads` | number of thread for transforming |
| `verbose` | whether to print the reminder |

## Value

number of individuals and markers. Output files: genotype.desc, genotype.bin: genotype file in bigmemory format phenotype.phe: ordered phenotype file, same taxa order with genotype file map.map: SNP information

## Examples

```
bfilePath <- file.path(system.file("extdata", "02_bfile", package = "rMVP"), "mvp")

MVP.Data.Bfile2MVP(bfilePath, tempfile("outfile"), threads=1)
```

MVP.Data.Hapmap2MVP *MVP.Data.Hapmap2MVP: To transform Hapmap data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018*

## Description

MVP.Data.Hapmap2MVP: To transform Hapmap data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.Hapmap2MVP(
  hmp_file,
  out = "mvp",
  maxLine = 10000,
  type.geno = "char",
  threads = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| hmp_file | Genotype in Hapmap format |
| out | the name of output file |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| type.geno | the type of genotype elements |
| threads | number of thread for transforming |
| verbose | whether to print the reminder |

## Value

number of individuals and markers. Output files: genotype.desc, genotype.bin: genotype file in bigmemory format phenotype.phe: ordered phenotype file, same taxa order with genotype file map.map: SNP information

## Examples

```
hapmapPath <- system.file("extdata", "03_hapmap", "mvp.hmp.txt", package = "rMVP")

MVP.Data.Hapmap2MVP(hapmapPath, tempfile("outfile"), threads=1)
```

`MVP.Data.impute` *MVP.Data.impute: To impute the missing genotype Author: Haohao Zhang Build date: Sep 12, 2018*

## Description

MVP.Data.impute: To impute the missing genotype Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.impute(
  mvp_prefix,
  out = NULL,
  mrk_bycol = TRUE,
  method = "Major",
  ncpus = NULL,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `mvp_prefix` | the prefix of mvp file |
| `out` | the prefix of output file |
| `mrk_bycol` | whether the markers are stored by columns in genotype (i.e. genotype is a n by m matrix) |
| `method` | 'Major', 'Minor', "Middle" |
| `ncpus` | number of threads for imputing |
| `verbose` | whether to print the reminder |

## Value

NULL Output files: imputed genotype file

## Examples

```
mvpPath <- file.path(system.file("extdata", "05_mvp", package = "rMVP"), "mvp")

MVP.Data.impute(mvpPath, tempfile("outfile"), ncpus=1)
```

MVP.Data.Kin *Kinship*

## Description

Kinship

## Usage

```
MVP.Data.Kin(
  fileKin = TRUE,
  mvp_prefix = "mvp",
  out = NULL,
  maxLine = 10000,
  mrk_bycol = TRUE,
  sep = "\t",
  cpu = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| fileKin | Kinship that represents relationship among individuals, n * n matrix, n is sample size |
| mvp_prefix | Prefix for mvp format files |
| out | prefix of output file name |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| mrk_bycol | whether the markers are stored by columns in genotype (i.e. genotype is a n by m matrix) |
| sep | seperator for Kinship file. |
| cpu | the number of cpu |
| verbose | whether to print detail. |

## Value

Output file: <out>.kin.bin <out>.kin.desc

## Examples

```
geno <- file.path(system.file("extdata", "06_mvp-impute", package = "rMVP"), "mvp.imp")

MVP.Data.Kin(TRUE, mvp_prefix=geno, out=tempfile("outfile"), cpu=1)
```

MVP.Data.Map *MVP.Data.Map: To check map file Author: Haohao Zhang Build date: Sep 12, 2018*

## Description

MVP.Data.Map: To check map file Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.Map(
  map,
  out = "mvp",
  cols = 1:5,
  header = TRUE,
  sep = "\t",
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| map | the name of map file or map object(data.frame or matrix) |
| out | the name of output file |
| cols | selected columns |
| header | whether the file contains header |
| sep | seperator of the file |
| verbose | whether to print detail. |

## Value

Output file: <out>.map

## Examples

```
mapPath <- system.file("extdata", "05_mvp", "mvp.geno.map", package = "rMVP")

MVP.Data.Map(mapPath, tempfile("outfile"))
```

`MVP.Data.MVP2Bfile` *MVP.Data.MVP2Bfile: To transform MVP data to binary format Author: Haohao Zhang Build date: Sep 12, 2018*

## Description

MVP.Data.MVP2Bfile: To transform MVP data to binary format Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.MVP2Bfile(
  bigmat,
  map,
  pheno = NULL,
  out = "mvp.plink",
  threads = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `bigmat` | Genotype in bigmatrix format (0,1,2) |
| `map` | the map file |
| `pheno` | the phenotype file |
| `out` | the name of output file |
| `threads` | number of thread for transforming |
| `verbose` | whether to print the reminder |

## Value

NULL Output files: .bed, .bim, .fam

## Examples

```
bigmat <- as.big.matrix(matrix(1:6, 3, 2))
map <- matrix(c("rs1", "rs2", "rs3", 1, 1, 1, 10, 20, 30), 3, 3)

MVP.Data.MVP2Bfile(bigmat, map, out=tempfile("outfile"), threads=1)
```

`MVP.Data.Numeric2MVP` *MVP.Data.Numeric2MVP: To transform Numeric data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018*

## Description

MVP.Data.Numeric2MVP: To transform Numeric data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.Numeric2MVP(
  num_file,
  map_file,
  out = "mvp",
  maxLine = 10000,
  row_names = FALSE,
  col_names = FALSE,
  type.geno = "char",
  auto_transpose = TRUE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `num_file` | Genotype in Numeric format (0,1,2) |
| `map_file` | Genotype map file, SNP_name, Chr, Pos |
| `out` | the name of output file |
| `maxLine` | the number of markers handled at a time, smaller value would reduce the memory cost |
| `row_names` | whether the numeric genotype has row names |
| `col_names` | whether the numeric genotype has column names |
| `type.geno` | the type of genotype elements |
| `auto_transpose` | whether to detecte the row and column |
| `verbose` | whether to print the reminder |

## Value

number of individuals and markers. Output files: genotype.desc, genotype.bin: genotype file in bigmemory format phenotype.phe: ordered phenotype file, same taxa order with genotype file map.map: SNP information

## Examples

```
numericPath <- system.file("extdata", "04_numeric", "mvp.num", package = "rMVP")
mapPath <- system.file("extdata", "04_numeric", "mvp.map", package = "rMVP")
MVP.Data.Numeric2MVP(numericPath, mapPath, tempfile("outfile"))
```

---

`MVP.Data.PC` *Principal component analysis*

---

## Description

Principal component analysis

## Usage

```
MVP.Data.PC(
  filePC = TRUE,
  mvp_prefix = "mvp",
  K = NULL,
  out = NULL,
  pcs.keep = 5,
  maxLine = 10000,
  mrk_bycol = TRUE,
  sep = "\t",
  cpu = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `filePC` | Principal components, n*npc, n is sample size, npc is number of top columns of principal components |
| `mvp_prefix` | Prefix for mvp format files |
| `K` | Kinship matrix |
| `out` | prefix of output file name |
| `pcs.keep` | how many PCs to keep |
| `maxLine` | the number of markers handled at a time, smaller value would reduce the memory cost |
| `mrk_bycol` | whether the markers are stored by columns in genotype (i.e. genotype is a n by m matrix) |
| `sep` | seperator for PC file. |
| `cpu` | the number of cpu |
| `verbose` | whether to print detail. |

## Value

Output file: <out>.pc.bin <out>.pc.desc

## Examples

```
geno <- file.path(system.file("extdata", "06_mvp-impute", package = "rMVP"), "mvp.imp")

MVP.Data.PC(TRUE, mvp_prefix=geno, out=tempfile("outfile"), cpu=1)
```

---

`MVP.Data.Pheno` *MVP.Data.Pheno: To clean up phenotype file Author: Haohao Zhang Build date: Sep 12, 2018*

---

## Description

MVP.Data.Pheno: To clean up phenotype file Author: Haohao Zhang Build date: Sep 12, 2018

## Usage

```
MVP.Data.Pheno(
  pheno_file,
  out = "mvp",
  cols = NULL,
  header = TRUE,
  sep = "\t",
  missing = c(NA, "NA", "-9", 9999),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `pheno_file` | the name of phenotype file |
| `out` | the name of output file |
| `cols` | selected columns |
| `header` | whether the file contains header |
| `sep` | seperator of the file |
| `missing` | the missing value |
| `verbose` | whether to print detail. |

## Value

NULL Output files: cleaned phenotype file

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")

MVP.Data.Pheno(phePath, out=tempfile("outfile"))
```

---

`MVP.Data.VCF2MVP` *MVP.Data.VCF2MVP: To transform vcf data to MVP package Author: Haohao Zhang Build date: Sep 12, 2018*

---

## Description

Accept the | or / separated markers, any variant sites that are not 0 or 1 will be considered NA.

## Usage

```
MVP.Data.VCF2MVP(
  vcf_file,
  out = "mvp",
  maxLine = 10000,
  type.geno = "char",
  threads = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `vcf_file` | Genotype in VCF format |
| `out` | the name of output file |
| `maxLine` | the number of markers handled at a time, smaller value would reduce the memory cost |
| `type.geno` | the type of genotype elements |
| `threads` | number of thread for transforming |
| `verbose` | whether to print the reminder |

## Value

number of individuals and markers. Output files: genotype.desc, genotype.bin: genotype file in bigmemory format phenotype.phe: ordered phenotype file, same taxa order with genotype file map.map: SNP information

## Examples

```
vcfPath <- system.file("extdata", "01_vcf", "mvp.vcf", package = "rMVP")

MVP.Data.VCF2MVP(vcfPath, tempfile("outfile"), threads=1)
```

`MVP.EMMA.Vg.Ve` *Estimate variance components using EMMA*

## Description

Build date: August 30, 2016 Last update: January 27, 2017

## Usage

```
MVP.EMMA.Vg.Ve(y, X, K, ngrids = 100, llim = -10, ulim = 10, esp = 1e-10)
```

## Arguments

| | |
|---|---|
| `y` | phenotype, n * 2 |
| `X` | covariate matrix, the first column is 1s |
| `K` | Kinship matrix |
| `ngrids` | parameters for estimating vg and ve |
| `llim` | parameters for estimating vg and ve |
| `ulim` | parameters for estimating vg and ve |
| `esp` | parameters for estimating vg and ve |

## Value

Output: REML - maximum log likelihood Output: delta - exp(root) Output: ve - residual variance Output: vg - genetic variance

## Author(s)

EMMA (Kang et. al. Genetics, 2008), Modified only for speed up by Xiaolei Liu and Lilin Yin

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
print(dim(genotype))

K <- MVP.K.VanRaden(genotype, cpu=1)
vc <- MVP.EMMA.Vg.Ve(y=phenotype[,2], X=matrix(1, nrow(phenotype)), K=K)
print(vc)
```

MVP.FarmCPU *Perform GWAS using FarmCPU method*

## Description

Date build: Febuary 24, 2013 Last update: May 25, 2017 Requirement: Y, GD, and CV should have same taxa order. GD and GM should have the same order on SNPs

## Usage

```
MVP.FarmCPU(
  phe,
  geno,
  map,
  CV = NULL,
  ind_idx = NULL,
  mrk_idx = NULL,
  P = NULL,
  method.sub = "reward",
  method.sub.final = "reward",
  method.bin = c("EMMA", "static", "FaST-LMM"),
  bin.size = c(5e+05, 5e+06, 5e+07),
  bin.selection = seq(10, 100, 10),
  memo = "MVP.FarmCPU",
  Prior = NULL,
  ncpus = 2,
  maxLoop = 10,
  maxLine = 5000,
  threshold.output = 0.01,
  converge = 1,
  iteration.output = FALSE,
  p.threshold = NA,
  QTN.threshold = 0.01,
  bound = NULL,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| phe | phenotype, n by t matrix, n is sample size, t is number of phenotypes |
| geno | genotype, either m by n or n by m is supportable, m is marker size, n is population size. This is Pure Genotype Data Matrix(GD). THERE IS NO COLUMN FOR TAXA. |
| map | SNP map information, m by 3 matrix, m is marker size, the three columns are SNP_ID, Chr, and Pos |
| CV | covariates, n by c matrix, n is sample size, c is number of covariates |

| | |
|---|---|
| ind_idx | the index of effective genotyped individuals |
| mrk_idx | the index of effective markers used in analysis |
| P | start p values for all SNPs |
| method.sub | method used in substitution process, five options: 'penalty', 'reward', 'mean', 'median', or 'onsite' |
| method.sub.final | method used in substitution process, five options: 'penalty', 'reward', 'mean', 'median', or 'onsite' |
| method.bin | method for selecting the most appropriate bins, three options: 'static', 'EMMA' or 'FaST-LMM' |
| bin.size | bin sizes for all iterations, a vector, the bin size is always from large to small |
| bin.selection | number of selected bins in each iteration, a vector |
| memo | a marker on output file name |
| Prior | prior information, four columns, which are SNP_ID, Chr, Pos, P-value |
| ncpus | number of threads used for parallele computation |
| maxLoop | maximum number of iterations |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| threshold.output | only the GWAS results with p-values lower than threshold.output will be output |
| converge | a number, 0 to 1, if selected pseudo QTNs in the last and the second last iterations have a certain probality (the probability is converge) of overlap, the loop will stop |
| iteration.output | whether to output results of all iterations |
| p.threshold | if all p values generated in the first iteration are bigger than p.threshold, FarmCPU stops |
| QTN.threshold | in second and later iterations, only SNPs with lower p-values than QTN.threshold have chances to be selected as pseudo QTNs |
| bound | maximum number of SNPs selected as pseudo QTNs in each iteration |
| verbose | whether to print detail. |

## Value

a m by 4 results matrix, m is marker size, the four columns are SNP_ID, Chr, Pos, and p-value

## Author(s)

Xiaolei Liu and Zhiwu Zhang

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
idx <- !is.na(phenotype[, 2])
phenotype <- phenotype[idx, ]
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
genotype <- deepcopy(genotype, rows=idx)
print(dim(genotype))
mapPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.map", package = "rMVP")
map <- read.table(mapPath , head = TRUE)

farmcpu <- MVP.FarmCPU(phe=phenotype,geno=genotype,map=map,maxLoop=2,method.bin="static")
str(farmcpu)
```

---

`MVP.FaSTLMM.LL` *Evaluation of the maximum likelihood using FaST-LMM method*

---

## Description

Last update: January 11, 2017

## Usage

```
MVP.FaSTLMM.LL(pheno, snp.pool, X0 = NULL, ncpus = 2)
```

## Arguments

| | |
|---|---|
| `pheno` | a two-column phenotype matrix |
| `snp.pool` | matrix for pseudo QTNs |
| `X0` | covariates matrix |
| `ncpus` | number of threads used for parallel computation |

## Value

Output: beta - beta effect Output: delta - delta value Output: LL - log-likelihood Output: vg - genetic variance Output: ve - residual variance

## Author(s)

Xiaolei Liu (modified)

MVP.GLM *To perform GWAS with GLM and MLM model and get the P value of SNPs*

## Description

Build date: Aug 30, 2016 Last update: May 25, 2017

## Usage

```
MVP.GLM(
  phe,
  geno,
  CV = NULL,
  ind_idx = NULL,
  mrk_idx = NULL,
  mrk_bycol = TRUE,
  maxLine = 5000,
  cpu = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| phe | phenotype, n * 2 matrix |
| geno | genotype, either m by n or n by m is supportable, m is marker size, n is population size |
| CV | Covariance, design matrix(n * x) for the fixed effects |
| ind_idx | the index of effective genotyped individuals |
| mrk_idx | the index of effective markers used in analysis |
| mrk_bycol | whether the markers are stored by columns in genotype (i.e. M is a n by m matrix) |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| cpu | number of cpus used for parallel computation |
| verbose | whether to print detail. |

## Value

m * 2 matrix, the first column is the SNP effect, the second column is the P values

## Author(s)

Lilin Yin and Xiaolei Liu

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
idx <- !is.na(phenotype[, 2])
phenotype <- phenotype[idx, ]
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
genotype <- deepcopy(genotype, rows=idx)
print(dim(genotype))

glm <- MVP.GLM(phe=phenotype, geno=genotype, cpu=1)
str(glm)
```

---

`MVP.HE.Vg.Ve` *To estimate variance component using HE regression*

---

## Description

Build date: Feb 2, 2017 Last update: Feb 2, 2019

## Usage

```
MVP.HE.Vg.Ve(y, X, K)
```

## Arguments

| | |
|---|---|
| y | phenotype |
| X | genotype |
| K | kinship matrix |

## Value

vg, ve, and delta

## Author(s)

Translated from C++(GEMMA, Xiang Zhou) to R by: Haohao Zhang

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
print(dim(genotype))
```

```
K <- MVP.K.VanRaden(genotype, cpu=1)
vc <- MVP.HE.Vg.Ve(y=phenotype[,2], X=matrix(1, nrow(phenotype)), K=K)
print(vc)
```

---

`MVP.Hist` *Phenotype distribution histogram*

---

## Description

Phenotype distribution histogram

## Usage

```
MVP.Hist(
  phe,
  col = c("dodgerblue4", "olivedrab4", "violetred", "darkgoldenrod1", "purple4"),
  breakNum = 15,
  memo = NULL,
  outpath = getwd(),
  test.method = "auto",
  file.type = "pdf",
  file.output = TRUE,
  dpi = 300
)
```

## Arguments

| | |
|---|---|
| phe | phenotype data |
| col | The color vector of the histogram. If the number of colors is less than break.n, the color will be reused. If the number of colors is greater than break.n, only the previous break.n colors will be used. |
| breakNum | the number of cells for the histogram. The default value is 15. |
| memo | Character. A text marker on output files |
| outpath | Effective only when file.output = TRUE, determines the path of the output file |
| test.method | The method used to test the normal distribution. The options are "auto", "Shapiro-Wilk", "Kolmogorov-Smirnov", and NULL. When set to "auto", "Shapiro- Wilk" method, "Kolmogorov-Smirnov" method will be used when it is greater than 5000, and it will not be tested when set to NULL. |
| file.type | A string or NULL is used to determine the type of output file. Can be "jpg", "pdf", "tiff". If it is NULL, it will use `dev.new()` to create a new graphics device in the current environment, which may be RStudioGD or the default device of the system. |
| file.output | Logical value. If TRUE, the figures will be generated. |
| dpi | The resolution of the image, specifying how many pixels per inch. |

## Value

Output file: MVP.Phe_Distribution.<trait>.<type>

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phe <- read.table(phePath, header=TRUE)

MVP.Hist(phe, file.output = FALSE)
```

---

`MVP.K.VanRaden` *Calculate Kinship matrix by VanRaden method*

---

## Description

Calculate Kinship matrix by VanRaden method

## Usage

```
MVP.K.VanRaden(
  M,
  maxLine = 5000,
  ind_idx = NULL,
  mrk_idx = NULL,
  mrk_freq = NULL,
  mrk_bycol = TRUE,
  cpu = 1,
  verbose = TRUE,
  checkNA = TRUE
)
```

## Arguments

| | |
|---|---|
| `M` | genotype, either m by n or n by m is supportable, m is marker size, n is population size |
| `maxLine` | the number of markers handled at a time, smaller value would reduce the memory cost |
| `ind_idx` | the index of effective genotyped individuals used in analysis |
| `mrk_idx` | the index of effective markers used in analysis |
| `mrk_freq` | the prior calculated major allele frequency (not MAF) for all markers used in analysis |
| `mrk_bycol` | whether the markers are stored by columns in genotype (i.e. M is a n by m matrix) |
| `cpu` | the number of cpu |
| `verbose` | whether to print detail. |
| `checkNA` | whether to check NA in genotype. |

## Value

K, n * n matrix

## Examples

```
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
print(dim(genotype))

K <- MVP.K.VanRaden(genotype, cpu=1)
```

---

MVP.MLM *To perform GWAS with GLM and MLM model and get the P value of SNPs*

---

## Description

To perform GWAS with GLM and MLM model and get the P value of SNPs

## Usage

```
MVP.MLM(
  phe,
  geno,
  K = NULL,
  eigenK = NULL,
  CV = NULL,
  ind_idx = NULL,
  mrk_idx = NULL,
  mrk_bycol = TRUE,
  REML = NULL,
  maxLine = 5000,
  cpu = 1,
  vc.method = c("BRENT", "EMMA", "HE"),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| phe | phenotype, n * 2 matrix |
| geno | genotype, either m by n or n by m is supportable, m is marker size, n is population size |
| K | Kinship, Covariance matrix(n * n) for random effects; must be positive semi-definite |
| eigenK | list of eigen Kinship |

| | |
|---|---|
| CV | covariates |
| ind_idx | the index of effective genotyped individuals |
| mrk_idx | the index of effective markers used in analysis |
| mrk_bycol | whether the markers are stored by columns in genotype (i.e. M is a n by m matrix) |
| REML | a list that contains ve and vg |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| cpu | number of cpus used for parallel computation |
| vc.method | the methods for estimating variance component("emma" or "he" or "brent") |
| verbose | whether to print detail. |

## Value

results: a m * 2 matrix, the first column is the SNP effect, the second column is the P values

## Author(s)

Lilin Yin and Xiaolei Liu

## Examples

```
phePath <- system.file("extdata", "07_other", "mvp.phe", package = "rMVP")
phenotype <- read.table(phePath, header=TRUE)
idx <- !is.na(phenotype[, 2])
phenotype <- phenotype[idx, ]
print(dim(phenotype))
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
genotype <- deepcopy(genotype, rows=idx)
print(dim(genotype))
K <- MVP.K.VanRaden(genotype, cpu=1)

mlm <- MVP.MLM(phe=phenotype, geno=genotype, K=K, cpu=1)
str(mlm)
```

---

# MVP.PCA *Principal Component Analysis*

---

## Description

Principal Component Analysis

## Usage

```
MVP.PCA(
  M = NULL,
  K = NULL,
  maxLine = 10000,
  ind_idx = NULL,
  mrk_idx = NULL,
  mrk_bycol = TRUE,
  pcs.keep = 5,
  cpu = 1,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| M | genotype, either m by n or n by m is supportable, m is marker size, n is population size |
| K | kinship matrix |
| maxLine | the number of markers handled at a time, smaller value would reduce the memory cost |
| ind_idx | the index of effective genotyped individuals used in analysis |
| mrk_idx | the index of effective markers used in analysis |
| mrk_bycol | whether the markers are stored by columns in genotype (i.e. M is a n by m matrix) |
| pcs.keep | maximum number of PCs for output |
| cpu | the number of cpu |
| verbose | whether to print detail. |

## Value

Output: PCs - a n * npc matrix of top number of PCs, n is population size and npc is @param pcs.keep

## Examples

```
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
genotype <- attach.big.matrix(genoPath)
print(dim(genotype))

pca <- MVP.PCA(M=genotype, cpu=1)
str(pca)
```

MVP.PCAplot *PCA Plot*

## Description

PCA Plot

## Usage

```
MVP.PCAplot(
  PCA,
  memo = "MVP",
  col = NULL,
  pch = NULL,
  class = NULL,
  legend.pos = "topright",
  Ncluster = 1,
  plot3D = FALSE,
  file.type = "pdf",
  dpi = 300,
  box = FALSE,
  file.output = TRUE,
  outpath = getwd(),
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| PCA | Principal component analysis result, 2-column matrix |
| memo | the prefix of the output image file. |
| col | colors for each cluster |
| pch | Either an integer specifying a symbol or a single character to be used as the default in plotting points. See points for possible values and their interpretation. Note that only integers and single-character strings can be set as a graphics parameter (and not NA nor NULL). |
| class | the class of all individuals, for example: "breed", "location" |
| legend.pos | position of legend. default is "topright" |
| Ncluster | cluster number |
| plot3D | (DEPRECATED)if TRUE, plot PC figure in 3D format, it can be only used in windows and mac operation system, "rgl" package should be installed beforehead |
| file.type | Character. Options are jpg, pdf, and tiff |
| dpi | Number. Dots per inch for .jpg and .tiff files |
| box | Logical value. If TRUE, the border line of Manhattan plot will be added |

| | |
|---|---|
| file.output | Logical value. If TRUE, the figures will be generated. |
| outpath | Effective only when file.output = TRUE, determines the path of the output file |
| verbose | whether to print detail. |

## Value

Output file: MVP.PCA_2D.<type>

## Examples

```
genoPath <- system.file("extdata", "06_mvp-impute", "mvp.imp.geno.desc", package = "rMVP")
geno <- attach.big.matrix(genoPath)
pca <- MVP.PCA(M=geno, cpu=1)

MVP.PCAplot(PCA=pca, Ncluster=3, class=NULL,
    col=c("red", "green", "yellow"), file.output=FALSE, pch=19)
```

---

MVP.Report *MVP.Report*

---

## Description

MVP.Report

## Usage

```
MVP.Report(
  MVP,
  col = c("#4197d8", "#f8c120", "#413496", "#495226", "#d60b6f", "#e66519", "#d581b7",
    "#83d3ad", "#7c162c", "#26755d"),
  bin.size = 1e+06,
  bin.range = NULL,
  pch = 19,
  band = 1,
  H = 1.5,
  ylim = NULL,
  cex.axis = 1,
  lwd.axis = 1.5,
  cex.lab = 1.5,
  plot.type = "b",
  multracks = FALSE,
  cex = c(0.5, 1, 1),
  r = 0.3,
  xlab = "Chromosome",
  ylab = expression(-log[10](italic(p))),
```

```
  xaxs = "i",
  yaxs = "r",
  outward = FALSE,
  threshold = NULL,
  threshold.col = "red",
  threshold.lwd = 1,
  threshold.lty = 2,
  amplify = FALSE,
  signal.cex = 1.5,
  signal.pch = 19,
  signal.col = "red",
  signal.line = 1,
  highlight = NULL,
  highlight.cex = 1.5,
  highlight.pch = 19,
  highlight.col = "green",
  chr.labels = NULL,
  chr.den.col = "black",
  cir.band = 1,
  cir.chr = TRUE,
  cir.chr.h = 1.5,
  cir.legend = TRUE,
  cir.legend.cex = 0.6,
  cir.legend.col = "black",
  LOG10 = TRUE,
  box = FALSE,
  conf.int = TRUE,
  file.output = TRUE,
  outpath = getwd(),
  file.type = "jpg",
  dpi = 300,
  height = NULL,
  width = NULL,
  memo = "",
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| MVP | a dataframe or list, at least four columns. The first column is the name of SNP, the second column is the chromosome of SNP, the third column is the position of SNP, and the remaining columns are the P-value of each trait(Note:each trait a column). |
| col | a vector or a matrix, if "col" is a vector, each circle use the same colors, it means that the same chromosome is drewed in the same color, the colors are not fixed, one, two, three or more colors can be used, if the length of the "col" is shorter than the length the chromosome, then colors will be applied circularly. If "col" is a matrix, the row is the number of circles(traits), the columns are the colors |

| | |
|---|---|
| | that users want to use for different circles, each circle can be plotted in different number of colors, the missing value can be replaced by NA. For example: col=matrix(c("grey30","grey60",NA,"red","blue","green","orange",NA,NA),3,3,byrow=T). |
| bin.size | the size of bin for SNP_density plot. |
| bin.range | a vector, c(min, max). The min/max value of legend of SNP_density plot, the bin whose SNP number is smaller/bigger than 'bin.range' will be use the same color. |
| pch | a number, the type for the points or for traits of multi-traits Manhattan plot, is the same with "pch" in <plot>. |
| band | a number, the space between chromosomes, the default is 1(if the band equals to 0, then there would be no space between chromosomes). |
| H | a number, the height for each circle, each circle represents a trait, the default is 1. |
| ylim | a vector, the range of Y-axis when plotting the two type of Manhattan plots, is the same with "ylim" in <plot>. |
| cex.axis | a number, controls the size of ticks' numbers of X/Y-axis and the size of labels of circle plot. |
| lwd.axis | a number, controls the width of X/Y-axis lines. |
| cex.lab | a number, controls the size of labels of X/Y-axis. |
| plot.type | a character or vector, only "d", "c", "m", "q" or "b" can be used. if plot.type="d", SNP density will be plotted; if plot.type="c", only circle-Manhattan plot will be plotted; if plot.type="m",only Manhattan plot will be plotted; if plot.type="q",only Q-Q plot will be plotted;if plot.type="b", both circle-Manhattan, Manhattan and Q-Q plots will be plotted; if plot.type=c("m","q"), Both Manhattan and Q-Q plots will be plotted. |
| multracks | a logical,if multracks=FALSE, all Manhattan plots will be drew in separated files, if it is TRUE, all Manhattan plots will be plotted in only one file. |
| cex | a number or a vector, the size for the points, is the same with "size" in <plot>, and if it is a vector, the first number controls the size of points in circle plot(the default is 0.5), the second number controls the size of points in Manhattan plot(the default is 1), the third number controls the size of points in Q-Q plot(the default is 1) |
| r | a number, the radius for the circle(the inside radius), the default is 1. |
| xlab | a character, the labels for x axis. |
| ylab | a character, the labels for y axis. |
| xaxs | a character, The style of axis interval calculation to be used for the x-axis. Possible values are "r", "i", "e", "s", "d". The styles are generally controlled by the range of data or xlim, if given. |
| yaxs | a character, The style of axis interval calculation to be used for the y-axis. See xaxs above.. |
| outward | logical, if outward=TRUE,then all points will be plotted from inside to outside for circular Manhattan plot. |

| | |
|---|---|
| threshold | a number or vector, the significant threshold. For example, Bonfferoni adjustment method: threshold=0.01/nrow(Pmap). More than one significant line can be added on the plots, if threshold=0 or NULL, then the threshold line will not be added. |
| threshold.col | a character or vector, the colour for the line of threshold levels. |
| threshold.lwd | a number or vector, the width for the line of threshold levels. |
| threshold.lty | a number or vector, the type for the line of threshold levels. |
| amplify | logical, CMplot can amplify the significant points, if amplify=T, then the points bigger than the minimal significant level will be amplified, the default: amplify=TRUE. |
| signal.cex | a number, if amplify=TRUE, users can set the size of significant points. |
| signal.pch | a number, if amplify=TRUE, users can set the shape of significant points. |
| signal.col | a character, if amplify=TRUE, users can set the colour of significant points, if signal.col=NULL, then the colors of significant points will not be changed. |
| signal.line | a number, the width of the lines of significant SNPs cross the circle. |
| highlight | a vector, names of SNPs which need to be highlighted. |
| highlight.cex | a number or vector, the size of points for SNPs which need to be highlighted. |
| highlight.pch | a number or vector, the pch of points for SNPs which need to be highlighted. |
| highlight.col | a number or vector, the col of points for SNPs which need to be highlighted. |
| chr.labels | a vector, the labels for the chromosomes of density plot and circle-Manhattan plot. |
| chr.den.col | a character or vector or NULL, the colour for the SNP density. If the length of parameter 'chr.den.col' is bigger than 1, SNP density that counts the number of SNP within given size('bin.size') will be plotted around the circle. If chr.den.col=NULL, the density bar will not be attached on the bottom of manhattan plot. |
| cir.band | a number, the space between circles, the default is 1. |
| cir.chr | logical, a boundary that represents chromosomes will be plotted on the periphery of a circle, the default is TRUE. |
| cir.chr.h | a number, the width for the boundary, if cir.chr=FALSE, then this parameter will be useless. |
| cir.legend | logical, whether to add the legend of each circle. |
| cir.legend.cex | a number, the size of the number of legend. |
| cir.legend.col | a character, the color of the axis of legend. |
| LOG10 | logical, whether to change the p-value into log10(p-value). |
| box | logical, this function draws a box around the current Manhattan plot. |
| conf.int | logical, whether to plot confidence interval on QQ-plot. |
| file.output | a logical, users can choose whether to output the plot results. |
| outpath | Only when file.output = TRUE, determines the path of the output file |
| file.type | a character, users can choose the different output formats of plot, so for, "jpg", "pdf", "tiff" can be selected by users. |

| | |
|---|---|
| dpi | a number, the picture resolution for .jpg and .tiff files. The default is 300. |
| height | the height of output files. |
| width | the width of output files. |
| memo | add a character to the output file name. |
| verbose | whether to print the reminder. |

## Value

Output files

## Examples

```
data(pig60K, package = "rMVP")

MVP.Report(pig60K[,c(1:3, 5)], plot.type="m",
    threshold=0.05/nrow(pig60K), file.output=FALSE)
```

---

`MVP.Report.Density` *SNP Density*

---

## Description

SNP Density

## Usage

```
MVP.Report.Density(
  Pmap,
  col = c("darkgreen", "yellow", "red"),
  dpi = 300,
  outpath = getwd(),
  memo = "MVP",
  bin.size = 1e+06,
  bin.max = NULL,
  file.type = "jpg",
  file.output = TRUE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| Pmap | P value Map |
| col | The color vector |
| dpi | Number. Dots per inch for .jpg and .tiff files |

| | |
|---|---|
| outpath | Only when file.output = TRUE, determines the path of the output file |
| memo | Character. A text marker on output files |
| bin.size | the window size for counting SNP number |
| bin.max | maximum SNP number, for winows, which has more SNPs than bin.max, will be painted in same color |
| file.type | format of output figure |
| file.output | Whether to output the file |
| verbose | whether to print detail. |

## Value

Output file: <memo>.SNP_Density.<type>

## Examples

```
data(pig60K, package = "rMVP")

MVP.Report.Density(pig60K, file.output=FALSE)
```

---

# MVP.Report.QQplot *QQ Plot*

---

## Description

QQ Plot

## Usage

```
MVP.Report.QQplot(
  P.values,
  taxa_name,
  col = c("blue"),
  cex = 0.5,
  threshold = NULL,
  amplify = TRUE,
  signal.col = "red",
  signal.pch = 19,
  signal.cex = 0.8,
  conf.int = TRUE,
  cex.axis = 1,
  conf.int.col = "grey",
  threshold.col = "red",
  outpath = getwd(),
  file.type = "jpg",
  memo = "MVP",
```

```
  box = TRUE,
  dpi = 300,
  file.output = TRUE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| P.values | P values |
| taxa_name | The identifier of the phenotype will be used to generate a portion of the image file name. If the title parameter is NULL, it will also be part of the title. |
| col | default color is "blue" |
| cex | A numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. This starts as 1 when a device is opened, and is reset when the layout is changed, e.g. by setting mfrow. see par. |
| threshold | Number or Vector. The cutoff line on Manhattan plot, e.g. Bonfferoni correction. More than one significant line can be added onto one figure. If threshold=0 or NULL, the threshold line will not be added. |
| amplify | Logical value. If TRUE, the points that passed the threshold line will be highlighted |
| signal.col | Character. If "amplify" is TRUE, "signal.col" is used to set the color of significant points, if "signal.col" is NULL, the colors of significant points will not be changed |
| signal.pch | Number. If "amplify" is TRUE, users can set the type of significant points |
| signal.cex | Number. If "amplify" is TRUE, "signal.cex" is used to set the size of significant points |
| conf.int | Whether to draw a confidence interval |
| cex.axis | a number, controls the size of numbers of X-axis and the size of labels of circle plot. |
| conf.int.col | a character, the color of the confidence interval on QQ-plot. |
| threshold.col | Character or Vector. The colors of threshold lines |
| outpath | Only when file.output = TRUE, determines the path of the output file |
| file.type | A string or NULL is used to determine the type of output file. Can be "jpg", "pdf", "tiff". If it is NULL, it will use dev.new() to create a new graphics device in the current environment, which may be RStudioGD or the default device of the system. |
| memo | the prefix of the output image file. |
| box | A Boolean value that controls whether to draw a box around QQplot. |
| dpi | a number, the picture element for .jpg and .tiff files. The default is 300. |
| file.output | Logical value. If TRUE, the figures will be generated. |
| verbose | whether to print detail. |

## Value

Output file: <memo>.QQplot.<taxa_name>.<type>

## Examples

```
data(pig60K, package = "rMVP")

MVP.Report(pig60K[1:10000,], plot.type="q", file.output=FALSE)
```

---

`MVP.Version` *Print MVP Banner*

---

## Description

Build date: Aug 30, 2017 Last update: Dec 12, 2018

## Usage

```
MVP.Version(width = 65, verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `width` | the width of the message |
| `verbose` | whether to print detail. |

## Value

version number.

## Author(s)

Lilin Yin, Haohao Zhang, and Xiaolei Liu

## Examples

```
MVP.Version()
```

pig60K *Genotyped by pig 60k chip*

## Description

This dataset gives the results of Genome-wide association study of 3 traits, individuals were genotyped by pig 60K chip.

## Usage

```
data(pig60K)
```

## Format

A dataframe containing 3 traits' Pvalue

# Index