Package: cause (via r-universe)

September 24, 2024

Type Package

Title CAUSE: Causal Analysis Using Summary Effect Estimates

Version 1.2.0.0335

Author Jean Morrison

Maintainer Jean Morrison <jvmorr@umich.edu>

Description Identify causal patterns using pairs of GWAS summary statistics.

License GPL (>= 2)

Encoding UTF-8

LazyData false

URL http://github.com/jean997/cause

BugReports http://github.com/jean997/cause/issues

RoxygenNote 7.1.1

LinkingTo Rcpp, RcppParallel

Imports Rcpp, RcppParallel, mixsqp (>= 0.3-43), numDeriv, ashr, dplyr, ggplot2, scales, gridExtra, loo (>= 2.3.0), matrixStats, purrr, tidyr, intervals, readr

SystemRequirements GNU make

Suggests testthat

Repository https://mrcieu.r-universe.dev

RemoteUrl https://github.com/jean997/cause

RemoteRef HEAD

RemoteSha 6257e6c66e6c1518c8750d1e4d2b4d6e42644fa5

Contents

`adapt2_grid` *Adaptive grid estimation of posteriors*

Description

Workhorse for posterior adaptive grid approximation. Called from cause_grid_adapt.

Usage

```
adapt2_grid(
  params,
  ranges,
  priors,
  n_start,
  range_fixed,
  mix_grid,
  rho,
  X,
  max_post_per_bin
)
```

`cause` *CAUSE*

Description

Fit CAUSE sharing and causal models, calculate ELPD test statistic and estimate posteriors

Implementation of CAUSE described in Morrison et al. (2020)

Usage

```
cause(
  X,
  param_ests,
  variants = X$snp,
  pval_thresh = 1,
  sigma_g,
  qalpha = 1,
  qbeta = 10,
  max_q = 1,
  force = FALSE,
  n_start_gamma_eta = 21,
  n_start_q = 11
)
```

Arguments

`X` An object of class cause_data containing data for the two traits.

`param_ests` Object of class cause_params output by est_cause_params. This contains estimates of the mixing proportions and an estimate of rho, the correlation in test statistics that is due to overlapping samples or population structure.

`variants` A vector of variants to include in the analysis.

`pval_thresh` Argument supplying the trait M p-value threshold for including a variant. If you would like to use all variants in ‘variants‘ without a threshold, use pval_thresh = 1.

`sigma_g` Parameter specifying the prior distribution of gamma and eta. gamma ~ N(0, sigma_g), eta ~ N(0, sigma_g).

`qalpha, qbeta` Parameters defining the prior distribution of q. q ~ Beta(qalpha, qbeta)

`max_q` Largest value of q to be allowed. If max_q < 1 then the prior will be truncated.

`force` If true, do not give an error if parameter estimates did not converge.

`n_start_gamma_eta, n_start_q`
Number of starting bins for grid approximation. You shouldn’t need to change these but if you are suspicious about your results, you might try increasing them. It’s best to use odd numbers.

Details

This function estimates posterior distributions for gamma, eta, and q under the sharing and causal models and computes a test statistic comparing the two models. The returned object contains

A note about arguments: The SNPs used to compute posteriors can be specified through two arguments, ‘variants‘ and ‘pval_thresh‘. ‘pval_thresh‘ was added in a later version and for compatibility with old code, the default value is set to 1. However, we recommend fitting posteriors with some p-value threshold in effect using either or both of the ‘variants‘ and ‘pval_thresh‘ arguments. The new argument makes it more convenient to do this without a separate step.

sharing, causal: posterior estimates under model.

elpd: A data frame giving estimated difference in elpd between the sharing and causal models. A negative delta_elpd favors the model in the "model 2" column. A positive delta_elpd favors the model in the "model 1" column.

data: The data used to compute the object. This data frame also contains posterior estimates for each variant of acting through U under both models.

sigma_g: Prior variance of gamma and eta. This value is chosen based on the data by default but can be supplied using the sigma_g parameter in the function call.

qalpha, qbeta: The prior distribution of q is Beta(qalpha, qbeta)

Functions summary() and plot() can be used with cause objects. See ?summary.cause and ?plot.cause

Value

An object of class "cause" that contains posterior estimates for causal and sharing models as well as results of the model comparison test. See Details.

Author(s)

Jean Morrison <jvmorr@umich.edu>

cause_download_example_gwas_data *Download GWAS data to use in examples*

Description

Download GWAS data to use in examples

Usage

```
cause_download_example_gwas_data()
```

Details

This function downloads Snakemake summary statistics for LDL cholesterol (Willer et al 2013 PMID 24097068), coronary artery disease (van der Harst et al 2017 PMID 29212778), and asthma (Demenais et al 2018 PMID 29273806) as well as a csv file that can be used with the CAUSE pipeline.

cause_grid_adapt *CAUSE posterior density estimation*

Description

Uses an adaptive grid approximation to estimate the posterior distribution of q, gamma, and eta or a subset of those

Usage

```
cause_grid_adapt(
  X,
  param_ests,
  params = c("q", "gamma", "eta"),
 priors = list(function(q) {    dbeta(q, 1, 10) }, function(b) {    dnorm(b, 0, 0.6)
    }, function(b) {     dnorm(b, 0, 0.6) }),
  n_start = c(10, 20, 20),
  max_post_per_bin = 0.001
)
```

Arguments

`X`	An object of class cause_data
`param_ests`	param_ests Object of class cause_params
`params`	Vector of parameter names. Should be some subest of c("q", "gamma", "eta") given in any order.

priors List of prior functions corresponding to parameters in params. Each element should be a funciton that takes one argument and returns a log likelihood.

n_start Vector of length length(params) giving the number of starting bins for each dimension

max_post_per_bin
Maximum posterior probability for each grid cube. This parameter controls the fineness of the approximation.

Value

An object of class cause_post.

est_cause_params *Estimate CAUSE Nuisance Parameters*

Description

Estimates bivariate distribution of summary statistics and rho, the correlation between summary statistics due to overlapping samples or population structure.

Usage

```
est_cause_params(
  X,
  variants,
  optmethod = c("mixSQP", "mixIP"),
  null_wt = 10,
  n_iter = 20,
  max_candidates = Inf,
  control = list()
)
```

Arguments

X An object of class cause_data containing data for the two traits.

variants A vector of variants to include. This list should be approximately LD pruned and include variants genome wide.

max_candidates Maximum number of variance candidates for each trait. Maximum size of grid is max_candidates^2

control control parameters to be passed to ashr. Primarily for development use, most users can leave this unaltered.

Value

An object of class cause_params

`est_cause_params_future` *Estimate CAUSE Nuisance Parameters*

Description

This is a development version of est_cause_params. It's performance is not as good as the exported function and we do not recomend that you use it. This function first estimates a causal effect using the modal estimator and then uses map_pi_rho_nonnull_eqg. Estimates bivariate distribution of summary statistics and rho, the correlation between summary statistics due to overlapping samples or population structure.

Usage

```
est_cause_params_future(
  X,
  variants,
  sigma_g,
  qalpha = 1,
  qbeta = 10,
  optmethod = c("mixSQP", "mixIP"),
  sigma_g_pval = 0.001,
  null_wt = 10,
  max_candidates = Inf
)
```

Arguments

`X`	An object of class cause_data containing data for the two traits.
`variants`	A vector of variants to include. This list should be approximately LD pruned and include variants genome wide.

Value

An object of class cause_params

`eta_gamma_prior` *Prior variance for gamma and eta*

Description

Choose prior variance for eta and gamma based on the data

Usage

```
eta_gamma_prior(X, variants, prob = 0.05, pval_thresh = 0.001)
```

Arguments

`X`	An object of class cause_data containing data for the two traits.
`variants`	A vector of variants to include in the analysis.
`prob`	See Details
`pval_thresh`	A p-value threshold applied to trait 1 (M). This prevents

Details

The function will return a variance, sigma, such that P(abs(z) > k) = prob where z is a N(0, sigma) random variable, k = max(abs(beta_hat_2/beta_hat_1)) and prob is specified in the arguments of the function.

Value

A prior variance for gamma and eta (scalar)

`gwas_format` *Convert GWAS summary statistics to a standard format*

Description

Format GWAS summary statistics for CAUSE

Usage

```
gwas_format(
  X,
  snp,
  beta_hat,
  se,
  A1,
  A2,
  chrom,
  pos,
  p_value,
  sample_size,
  output_file,
  compute_pval = TRUE
)
```

Arguments

`X`	data.frame
`snp`	Column name containing SNP ID
`beta_hat`	Column name containing effect estimate

`se` Column name containing standard error of beta_hat

`A1` Column name containing effect allele

`A2` Column name containing other allele

`chrom` Chromosome column (optional)

`pos` Position column (optional)

`p_value` p-value column (optional)

`sample_size` Sample size column (optional) or an integer

`output_file` File to write out formatted data. If missing formatted data will be returned.

`compute_pval` Logical, compute the p-value using a normal approximation if missing? Defaults to TRUE.

Details

This function will try to merge data sets X1 and X2 on the specified columns. Where necessary, it will flip the sign of effects so that the effect allele is the same in both data sets. It will remove variants with ambiguous alleles or where the alleles (G/C or A/T) or with alleles that do not match between data sets (e.g A/G in one data set and A/C in the other). It will not remove variants that are simply strand flipped between the two data sets (e. g. A/C in one data set, T/G in the other).

Value

A data frame with columns chrom, pos, snp, A1, A2, beta_hat, se, p_value, and sample_size with all SNPs aligned so that A is the effect allele. This is ready to be used with gwas_merge with formatted = TRUE.

`gwas_merge` *Format data for CAUSE*

Description

Format GWAS summary statistics for CAUSE

Usage

```
gwas_merge(
  X1,
  X2,
  X1_formatted = FALSE,
  X2_formatted = FALSE,
  snp_name_cols = c("snp", "snp"),
  beta_hat_cols = c("beta_hat", "beta_hat"),
  se_cols = c("se", "se"),
  A1_cols = c("A1", "A1"),
  A2_cols = c("A2", "A2"),
  pval_cols = c(NA, NA),
  compute_pvals = TRUE
)
```

Arguments

X1 data.frame with data for GWAS 1

X2 data.frame with data for GWAS 2

snp_name_cols A vector of length 2 specifying the name of the snp column in X1 and X2 respectively. This is the column on which the data will be merged.

beta_hat_cols A vector of length 2 specifying the name of the snp column in X1 and X2 respectively. If effect sizes are provided as odds ratios they must be converted back into coeffecient estimates by taking the log.

se_cols A vector of length 2 specifying the name of the snp column in X1 and X2 respectively.

A1_cols Column names for effect allele

A2_cols Column names for other allele

pval_cols Column names for p-values. Can be ommitted or either element can be NA.

compute_pvals If p-values are missing in one or both studies, should the function compute them. If true, p-values will be computed as ‘2*pnorm(-abs(beta_hat/se))‘.

Details

This function will try to merge data sets X1 and X2 on the specified columns. Where necessary, it will flip the sign of effects so that the effect allele is the same in both data sets. It will remove variants with ambiguous alleles or where the alleles (G/C or A/T) or with alleles that do not match between data sets (e.g A/G in one data set and A/C in the other). It will not remove variants that are simply strand flipped between the two data sets (e. g. A/C in one data set, T/G in the other).

Value

An object of class cause_data and data.frame.

in_sample_elpd_loo *Estimate delta elpd for pairs of models*

Description

Estimate delta elpd for pairs of models

Usage

```
in_sample_elpd_loo(X, fits, variants, nsamps = 1000)
```

Arguments

X Data

fits List of models of class cause_post.

variants Optional vector of SNPs to use for ELPD calculation

nsamps Number of samples to take from the posterior

Value

A data frame with one row per pair of models giving the estimated difference in elpds and standard error of that difference. If delta_elpd is negative this is in favor of model 2 in the table. A positive delta_elpd is in favor of model 1.

---

ld_prune *Prune variants for LD*

Description

Get an LD pruned list of variants optionally prioritizing variants with low p-values

Usage

```
ld_prune(
  variants,
  ld,
  total_ld_variants,
  pval_cols,
  pval_thresh,
  r2_thresh = 0.1,
  variant_name = "snp"
)
```

Arguments

variants	Either a vector of SNP names or a data.frame. If a data.frame, provide pval_cols containing the names of pvalue columns to use for pruning. Results will contain one list per p-value column. If variants is a vector, LD pruning will be random.
ld	data.frame with three columns: colsnp, rowsnp, and r2. Each row corresponds to an entry of the LD matrix. If a pair of variants does not appear, they are assumed to have LD of 0 (or below the threshold).
total_ld_variants	vector of variant names used to construct 'ld'.
pval_cols	Vector of names of pval_cols. If not missing, variants must be a data.frame. pval_cols may contain NA entries. NA's indicate that you desire a pruned set that is not prioritized by p-value.
pval_thresh	A vector the same length as 'pval_cols'. Only variants with p-value below the threshold will be retained. Not required if variants is a vector. Entries of pval_thresh may be Inf.
r2_thresh	r^2 threshold for pruning. The final list will contain no pairs with r2 > r2_thresh
variant_name	If variants is a data.frame, the column name containing the variant id.

Details

ld_prune will generate any desired number of LD pruned variant lists. At minimum, it requires three arguments: 'variants' is the set of variants to be pruned, 'ld' is a data frame giving estimated LD from a reference panel and 'total_ld_variants' is the list of variants used to construct the ld data. Only variants ovlapping between 'variants' and 'total_ld_variants' will be considered.

If only these three arguments are given, the returned object will be a vector of LD pruned variants. These will be pruned randomly as no p-value information has been supplied.

Another option is to prune variants prefentially retaining those with small p-values. For this option, the 'variants' argument should be a data.frame that includes at minimum a column containing the variant name and a column containing the pvalue. You can prune on multiple p-values at once which can save time.

loglik *Calculate likelihood – version 7*

Description

Calculate likelihood – version 7

Usage

```
loglik(
  rho,
  g,
  gp,
  q,
  tsigma1,
  tsigma2,
  tpi,
  tbeta_hat_1,
  tbeta_hat_2,
  tseb1,
  tseb2
)
```

loglik_ij *Calculate likelihood of (beta_hat_1[j], beta_hat_2[j])) given rho, b, sigma_1j, sigma_2j, s_1j, s_2j where alpha_kj ~ N(0, sigma_kj)*

Description

Calculate likelihood of (beta_hat_1[j], beta_hat_2[j])) given rho, b, sigma_1j, sigma_2j, s_1j, s_2j where alpha_kj ~ N(0, sigma_kj)

Usage

```
loglik_ij(rho, g, gp, q, sigma1, sigma2, b1, b2, s1, s2)
```

`loglik_loo` *Calculate likelihood matrix for LOO (SNPs by posterior samples)*

Description

Calculate likelihood matrix for LOO (SNPs by posterior samples)

Usage

```
loglik_loo(
  tg,
  tgp,
  tq,
  rho,
  tsigma1,
  tsigma2,
  tpi,
  tbeta_hat_1,
  tbeta_hat_2,
  tseb1,
  tseb2
)
```

Value

A matrix that is p by L where p is the number of SNPs and L is the number of posterior samples

`loglik_mat` *Calculate likelihood matrix – version 7*

Description

Calculate likelihood matrix – version 7

Usage

```
loglik_mat(
  rho,
  g,
  gp,
  q,
  tsigma1,
  tsigma2,
  tbeta_hat_1,
  tbeta_hat_2,
  tseb1,
  tseb2
)
```

Value

A matrix that is p by K where p is the number of SNPs and K is the grid size

---

`loglik_samps_Z0` *Calculate likelihood matrix for LOO (SNPs by posterior samples)*

Description

Calculate likelihood matrix for LOO (SNPs by posterior samples)

Usage

```
loglik_samps_Z0(
  tg,
  tgp,
  tq,
  rho,
  tsigma1,
  tsigma2,
  tpi,
  tbeta_hat_1,
  tbeta_hat_2,
  tseb1,
  tseb2
)
```

Value

A matrix that is p by L where p is the number of SNPs and L is the number of posterior samples

loglik_samps_Z1 *Calculate likelihood matrix for LOO (SNPs by posterior samples)*

Description

Calculate likelihood matrix for LOO (SNPs by posterior samples)

Usage

```
loglik_samps_Z1(
  tg,
  tgp,
  tq,
  rho,
  tsigma1,
  tsigma2,
  tpi,
  tbeta_hat_1,
  tbeta_hat_2,
  tseb1,
  tseb2
)
```

Value

A matrix that is p by L where p is the number of SNPs and L is the number of posterior samples

map_pi_rho *Estimate pi and mixture proportions under null model*

Description

This function is called by est_cause_params and generally does not need to be directly accessed. Estimate the MAP for rho and the mixing proportions using coordinate descent. Causal effect (gamma) and the effect of U (eta) are fixed at zero.

Usage

```
map_pi_rho(
  X,
  mix_grid,
  rho_start = 0,
  tol = 1e-07,
  n_iter = 20,
  null_wt = 10,
  z_prior_func = function(z) {     dnorm(z, 0, 0.5, log = TRUE) },
  optmethod = c("mixSQP", "mixIP"),
  control = list(),
  warm = FALSE
)
```

Arguments

X	An object of class cause_data containing data for the two traits.
mix_grid	An object of class cause_grid containing variance pair candidates
rho_start	Starting value for rho
null_wt	Specifies the prior weight on the first entry of grid
z_prior_func	Prior function for z = arctanh(rho)

map_pi_rho_nonnull *Estimate pi and mixture proportions under a specified model*

Description

Currently this function is not exported and is mostly for development purposes. It is just like map_pi_rho except that one can specify gamma, eta, and q rather than having them fixed at zero.

Usage

```
map_pi_rho_nonnull(
  X,
  mix_grid,
  sigma_g,
  qalpha,
  qbeta,
  q_start = qbeta(0.5, qalpha, qbeta),
  gamma_start = 0,
  eta_start = 0,
  rho_start = 0,
  tol = 1e-05,
  n.iter = 20,
  null_wt = 10,
  z_prior_func = function(z) {     dnorm(z, 0, 0.5, log = TRUE) },
  q_prior_func = function(q) {     dbeta(q, qalpha, qbeta, log = TRUE) },
  gammaeta_prior_func = function(g) {     dnorm(g, 0, sigma_g, log = TRUE) },
  optmethod = c("mixSQP", "mixIP")
)
```

Arguments

`X`	An object of class cause_data containing data for the two traits.
`mix_grid`	An object of class cause_grid containing variance pair candidates
`rho_start`	Starting value for rho
`null_wt`	Specifies the prior weight on the first entry of grid
`z_prior_func`	Prior function for z = arctanh(rho)

---

map_pi_rho_nonnull_eqg

Estimate pi and mixture proportions under a specified model

Description

Estimare the MAP for rho and the mixing proportions using coordinate descent. Causal effect (gamma) and the effect of U (eta) are fixed at zero.

Usage

```
map_pi_rho_nonnull_eqg(
  X,
  mix_grid,
  sigma_g,
  qalpha,
  qbeta,
  q_start = qbeta(0.5, qalpha, qbeta),
  gamma_start = 0,
  eta_start = 0,
  rho_start = 0,
  tol = 1e-05,
  n.iter = 20,
  null_wt = 10,
  z_prior_func = function(z) {     dnorm(z, 0, 0.5, log = TRUE) },
  q_prior_func = function(q) {     dbeta(q, qalpha, qbeta, log = TRUE) },
  gammaeta_prior_func = function(g) {     dnorm(g, 0, sigma_g, log = TRUE) },
  optmethod = c("mixSQP", "mixIP")
)
```

Arguments

`X`	An object of class cause_data containing data for the two traits.
`mix_grid`	An object of class cause_grid containing variance pair candidates
`rho_start`	Starting value for rho
`null_wt`	Specifies the prior weight on the first entry of grid
`z_prior_func`	Prior function for z = arctanh(rho)

map_pi_rho_nonnull_fixed

Estimate pi and mixture proportions under a specified model

Description

Estimare the MAP for rho and the mixing proportions using coordinate descent. Causal effect (gamma) and the effect of U (eta) are fixed at zero.

Usage

```
map_pi_rho_nonnull_fixed(
  X,
  mix_grid,
  gamma,
  eta,
  q,
  rho_start = 0,
  tol = 1e-05,
  n.iter = 20,
  null_wt = 10,
  z_prior_func = function(z) {     dnorm(z, 0, 0.5, log = TRUE) },
  optmethod = c("mixSQP", "mixIP")
)
```

Arguments

`X`	An object of class cause_data containing data for the two traits.
`mix_grid`	An object of class cause_grid containing variance pair candidates
`rho_start`	Starting value for rho
`null_wt`	Specifies the prior weight on the first entry of grid
`z_prior_func`	Prior function for z = arctanh(rho)

map_pi_rho_nonnull_geq

Estimate pi and mixture proportions under a specified model

Description

Estimare the MAP for rho and the mixing proportions using coordinate descent. Causal effect (gamma) and the effect of U (eta) are fixed at zero.

Usage

```
map_pi_rho_nonnull_geq(
  X,
  mix_grid,
  sigma_g,
  qalpha,
  qbeta,
  q_start = qbeta(0.5, qalpha, qbeta),
  gamma_start = 0,
  eta_start = 0,
  rho_start = 0,
  tol = 1e-05,
  n.iter = 20,
  null_wt = 10,
  z_prior_func = function(z) {     dnorm(z, 0, 0.5, log = TRUE) },
  q_prior_func = function(q) {     dbeta(q, qalpha, qbeta, log = TRUE) },
  gammaeta_prior_func = function(g) {     dnorm(g, 0, sigma_g, log = TRUE) },
  optmethod = c("mixSQP", "mixIP")
)
```

Arguments

`X` An object of class cause_data containing data for the two traits.

`mix_grid` An object of class cause_grid containing variance pair candidates

`rho_start` Starting value for rho

`null_wt` Specifies the prior weight on the first entry of grid

`z_prior_func` Prior function for z = arctanh(rho)

`map_pi_rho_nonnull_qge` *Estimate pi and mixture proportions under a specified model*

Description

Estimare the MAP for rho and the mixing proportions using coordinate descent. Causal effect (gamma) and the effect of U (eta) are fixed at zero.

Usage

```
map_pi_rho_nonnull_qge(
  X,
  mix_grid,
  sigma_g,
  qalpha,
  qbeta,
  q_start = qbeta(0.5, qalpha, qbeta),
  gamma_start = 0,
  eta_start = 0,
  rho_start = 0,
  tol = 1e-05,
  n.iter = 20,
  null_wt = 10,
  z_prior_func = function(z) {     dnorm(z, 0, 0.5, log = TRUE) },
  q_prior_func = function(q) {     dbeta(q, qalpha, qbeta, log = TRUE) },
  gammaeta_prior_func = function(g) {     dnorm(g, 0, sigma_g, log = TRUE) },
  optmethod = c("mixSQP", "mixIP")
)
```

Arguments

`X` An object of class cause_data containing data for the two traits.

`mix_grid` An object of class cause_grid containing variance pair candidates

`rho_start` Starting value for rho

`null_wt` Specifies the prior weight on the first entry of grid

`z_prior_func` Prior function for z = arctanh(rho)

`new_cause_data` *Create a new cause_data object*

Description

Create a new cause_data object

Usage

```
new_cause_data(x = data.frame())
```

Arguments

`x` a data.frame that includes columns snp, beta_hat_1, seb1, beta_hat_2, seb2 in any order. x may also contain other columns

Value

and object of class cause_data and data.frame.

new_cause_data_fit *Create a new cause_data_fit object*

Description

Create a new cause_data_fit object

Usage

```
new_cause_data_fit(x = data.frame())
```

Arguments

x	a data.frame that includes columns snp, beta_hat_1, seb1, beta_hat_2, seb2, delta_elpd, prob_Z1_sharing, prob_Z1_causal in any order. x may also contain other columns

Value

and object of class cause_data and data.frame.

plot.cause *Plot CAUSE*

Description

Plot posteriors for sharing and causal models, display summary tables

Usage

```
## S3 method for class 'cause'
plot(res, intern = FALSE, type = c("posteriors", "data"), pval_thresh = 5e-08)
```

Arguments

res	object of class cause
intern	If TRUE, function returns a list of grobs. Otherwise it plots.
type	Either "posteriors" or "data". See details.
pval_thresh	p-value threshold used if type=="data".

Details

If type == "posteriors", the function will plot the posterior distributions of the parameters and display summary tables giving medians and credible intervals. If type == "data" the function will plot the data thresholded on the trait 1 p-value if using pval_thresh.

plot.cause_post *Plot posteriors for one CAUSE fit*

Description

Plot posteriors for one CAUSE fit

Usage

```
## S3 method for class 'cause_post'
plot(fit, type = c("posteriors", "data"), data = NULL, pval_thresh = 5e-08)
```

Arguments

`fit`	object of class cause_post ‘
`type`	Either "posteriors" or "data". See details.
`data`	If type=="data" then an object of type cause_data_fit must be supplied
`pval_thresh`	p-value threshold used when type=="data".

Details

If type == "posteriors", the function plots the marginal posterior distribution of each parameter. If type=="data", the function plots data colored by the probability that Z = 1.

print.cause_summary *Print CAUSE Summary*

Description

Print a CAUSE summary

Usage

```
## S3 method for class 'cause_summary'
print(x, digits = 2)
```

Arguments

`x`	object of class cause_summary
`digits`	significant digits to report.

print.summary_cause_post *Print CAUSE fit Summary*

Description

Print a CAUSE fit summary

Usage

```
## S3 method for class 'summary_cause_post'
print(x)
```

Arguments

`x`	object of class cause_summary_post

recompute_elpd_table *Recompute elpd table for a CAUSE fit that is already computed*

Description

Recompute elpd table for a CAUSE fit that is already computed

Usage

```
recompute_elpd_table(res)
```

Arguments

`res`	A cause fit from running the cause function

Value

A model table equivalent to res$elpd

samp_from_grid *Sample from grid approximation of posterior distribuitions*

Description

Sample from the joint posterior of parameters

Usage

```
samp_from_grid(joint_post, params, nsamps)
```

Arguments

joint_post	Data frame giving the joint posterior
params	Vector of names of parameters to sample
nsamps	Number of samples to take

Value

A data.frame that is nsamps by length(params) of samples

setup_cause_pipeline *Set up analysis directory for pairs of GWAS summary statistics*

Description

A helper function that prepares a working directory for running an analysis with CAUSE.

Usage

```
setup_cause_pipeline(download_ldshrink = FALSE, download_eur_ld_scores = FALSE)
```

Arguments

download_ldshrink	If TRUE the function will download LDshrink LD estimates for european subset of 1k genomes
download_eur_ld_scores	If TRUE the function will download LD scores needed for running LCV.

Details

This function downloads Snakemake pipeline and R code needed to run CAUSE on many pairs of traits.

`setup_gwas_pairs` *Set up analysis directory for pairs of GWAS summary statistics*

Description

A helper function that prepares a working directory for running an analysis with CAUSE.

Usage

```
setup_gwas_pairs()
```

Details

This function prepares a working directory to replicate the results in Section 2.3 of Morrison et al 2019 (https://www.biorxiv.org/content/10.1101/682237v3). For more details see https://jean997.github.io/cause/gwas_pairs.h

`step_quantile` *Quantile of a step pdf*

Description

Quantile of a step pdf

Usage

```
step_quantile(x, start, stop, p)
```

Arguments

`x`	quantile or vector of quantiles
`start`	vector of step starts
`stop`	vector of step stops
`p`	vector of step probabilities

summary.cause *Summarize CAUSE Results*

Description

Summarize p-value testing that the causal model fits the data better and the posterior medians and credible intervals for parameters

Usage

```
## S3 method for class 'cause'
summary(res, ci_size = 0.95, digits = 2)
```

Arguments

res	object of class cause.
ci_size	size of posterior credible intervals.
digits	significant digits to report.

summary.cause_post *Summarize CAUSE Results for a single fit*

Description

Summarize posterior medians and credible intervals for parameters

Usage

```
## S3 method for class 'cause_post'
summary(fit, ci_size = 0.95, digits = 2)
```

Arguments

fit	object of class cause_post.
ci_size	size of posterior credible intervals.
digits	significant digits to report.

Z_post *Posterior probability of acting through U*

Description

For a set of data, X, calculate the probability that each variant acts through the shared factor U

Usage

```
Z_post(X, fit, nsamps = 1000, name_post = "")
```

Arguments

`X`	object of class cause_data. This need not be the same data used to compute 'fit'.
`fit`	object of class cause_post.
`nsamps`	Number of samples to take from the posterior (see Details).

Details

Let x_i represent the the data in the ith row of 'X'. Let Z_i be the indicator that variant i affects U. We are interested in computing P(Z_i = 1 | posteriors) where posteriors refers to the posterior parameter distributions contained in 'fit'. We compute l(x_i | posteriors, Z=1) and l(x_i | posteriors, Z_i = 0) where l denotes the likelihood. We then compute P(Z_i = 1 | posteriors) = l(x_i | posteriors, Z_i = 1)/(l(x_i | posteriors, Z_i = 1) + l(x_i | posteriors, Z_i = 0))

Value

A vector of probabilities corresponding to the rows of X

Index