

Package: susieR (via r-universe)

October 11, 2026

Encoding UTF-8

Type Package

Title Sum of Single Effects Linear Regression

Description Implements methods for variable selection in linear regression based on the ``Sum of Single Effects" (SuSiE) model, as described in Wang et al (2020) <[DOI:10.1101/501114](https://doi.org/10.1101/501114)> and Zou et al (2021) <[DOI:10.1101/2021.11.03.467167](https://doi.org/10.1101/2021.11.03.467167)>. These methods provide simple summaries, called ``Credible Sets", for accurately quantifying uncertainty in which variables should be selected. The methods are motivated by genetic fine-mapping applications, and are particularly well-suited to settings where variables are highly correlated and detectable effects are sparse. The fitting algorithm, a Bayesian analogue of stepwise selection methods called ``Iterative Bayesian Stepwise Selection" (IBSS), is simple and fast, allowing the SuSiE model be fit to large data sets (thousands of samples and hundreds of thousands of variables).

Date 2026-07-28

Version 0.16.6

URL <https://github.com/stephenslab/susieR>

BugReports <https://github.com/stephenslab/susieR/issues>

License BSD_3_clause + file LICENSE

Depends R (>= 3.0.2)

Imports methods, graphics, grDevices, stats, Matrix, matrixStats, mixsqp, reshape, crayon, ggplot2, Rfast

LinkingTo cpp11, cpp11armadillo

Suggests curl, pkgload, rprojroot, testthat, microbenchmark, knitr, rmarkdown, cowplot, L0Learn

LazyData yes

LazyDataCompression xz

NeedsCompilation yes

RoxygenNote 7.3.3
VignetteBuilder knitr
Config/pak/sysreqs cmake
Repository <https://stephenslab.r-universe.dev>
Date/Publication 2026-10-11 20:49:26 UTC
RemoteUrl <https://github.com/stephenslab/susier>
RemoteRef HEAD
RemoteSha 69db7e7a3f917ea02a4abc4b53694b2cd1c26b7e

Contents

absolute.order	3
block_coordinate_ascent	4
coef.mr.ash	5
coef.susie	5
compute_marginal_bhat_shat	6
compute_suff_stat	7
data_small	8
estimate_s_rss	8
FinemappingConvergence	9
get.full.posterior	10
get_cs_correlation	11
kriging_rss	12
mr.ash	13
mr.ash.rss	17
N2finemapping	19
N3finemapping	20
path.order	21
predict.mr.ash	22
predict.susie	23
print.summary.susie_post_outcome_configuration	23
rss_mismatch_cases	24
slot_prior_betabinom	25
summary.susie	27
summary.susie_post_outcome_configuration	27
SummaryConsistency	29
susie	29
susie_auto	35
susie_get_objective	37
susie_init_coef	40
susie_plot	41
susie_plot_changepoint	43
susie_post_outcome_configuration	44
susie_rss	47
susie_rss_control	53

<i>absolute.order</i>	3
susie_rss_lambda	54
susie_ser	58
susie_ss	60
susie_trendfilter	64
univar.order	66
univariate_regression	67
unmappable_data	68
Index	69

<i>absolute.order</i>	<i>Ordering of Predictors from Coefficient Estimates</i>
-----------------------	--

Description

This function orders the predictors by decreasing order of the magnitude of the estimated regression coefficient.

Usage

`absolute.order(beta)`

Arguments

`beta` A vector of estimated regression coefficients.

Value

An ordering of the predictors.

Examples

```
### generate synthetic data
set.seed(1)
n      = 200
p      = 300
X      = matrix(rnorm(n*p),n,p)
beta   = double(p)
beta[1:10] = 1:10
y      = X %*% beta + rnorm(n)

### order predictors by magnitude of univariate regression coefficient
beta.hat = univariate_regression(X,y)$betahat
order    = absolute.order(beta.hat)
```

block_coordinate_ascent

Block coordinate ascent for iterative model refinement.

Description

Generic framework for post-convergence refinement of fitted models. Repeatedly applies a block update (`step_fn`) to a fitted model, monitoring ELBO for convergence and stability. Both CS refinement (`run_refine`) and residual variance estimation (`mvsusieR`) are instances of block coordinate ascent over different parameter blocks.

Usage

```
block_coordinate_ascent(
  model,
  data,
  step_fn,
  max_iter = 100,
  tol = 0.001,
  verbose = FALSE
)
```

Arguments

<code>model</code>	Fitted model (e.g., from <code>susie_workhorse</code> or <code>mvsusie_workhorse</code>).
<code>data</code>	Data object passed to <code>step_fn</code> .
<code>step_fn</code>	A function with signature <code>function(model, data, iter)</code> that performs one block coordinate update. Must return a named list with elements: model (required) The updated model. data (optional) Updated data object, e.g. after changing residual variance. If <code>NULL</code>, the data is unchanged. converged (optional) Logical; if <code>TRUE</code>, stop iterating. log_msg (optional) Character string appended to verbose output.
<code>max_iter</code>	Maximum number of block ascent iterations (default 100).
<code>tol</code>	Convergence tolerance for relative ELBO change (default 1e-3).
<code>verbose</code>	If <code>TRUE</code> , print progress each iteration (default <code>FALSE</code>).

Details

Convergence is reached when either:

- `step_fn` returns `converged = TRUE` (the block update signals no further improvement), or
- the relative ELBO change falls below `tol` (ELBO stabilized across block updates).

A warning is issued if the ELBO decreases between iterations.

Value

The refined model, with `model$converged` set to TRUE or FALSE.

`coef.mr.ash`
Extract Regression Coefficients from Mr.ASH Fit

Description

Retrieve posterior mean estimates of the regression coefficients in a Mr.ASH model.

Usage

```
## S3 method for class 'mr.ash'
coef(object, ...)
```

Arguments

`object` A Mr.ASH fit, usually the result of calling `mr.ash`.
`...` Additional arguments passed to the default S3 method.

Value

A $p+1$ vector. The first element gives the estimated intercept, and the remaining p elements are the estimated regression coefficients.

```
## generate synthetic data set.seed(1) n = 200 p = 300 X = matrix(rnorm(n*p),n,p) beta = double(p)
beta[1:10] = 1:10 y = X
```

```
## fit mr.ash fit.mr.ash = mr.ash(X, y)
```

```
## coefficient coef.mr.ash = coef(fit.mr.ash) intercept = coef.mr.ash[1] beta = coef.mr.ash[-1]
```

`coef.susie`
Extract regression coefficients from susie fit

Description

Extract regression coefficients from susie fit

Usage

```
## S3 method for class 'susie'
coef(object, ...)
```

Arguments

`object` A susie fit.
`...` Additional arguments passed to the generic `coef` method.

Value

A $p+1$ vector, the first element being an intercept, and the remaining p elements being estimated regression coefficients.

compute_marginal_bhat_shat

Per-Position Marginal OLS Regression of ‘Y’ on Each Column of ‘X’

Description

Computes the marginal OLS regression coefficient and standard error for each ‘(X column, Y column)’ pair, treating the regressions as independent. ‘X’ is assumed column-centred (no intercept term in the per-pair regression); each ‘Y’ column is treated independently. Returns the $J \times T$ matrices ‘Bhat’ and ‘Shat’.

Used internally by single-effect-regression style routines that need a per-position marginal estimate. Vectorised across columns of ‘Y’ so callers can pass either a numeric vector ($T = 1$) or a numeric matrix ($T > 1$) without looping at the call site.

Usage

```
compute_marginal_bhat_shat(X, Y, predictor_weights = NULL, sigma2 = NULL)
```

Arguments

X	numeric matrix ‘ $n \times J$ ’, expected column-centred.
Y	numeric matrix ‘ $n \times T$ ’ or numeric vector of length ‘ n ’. When a vector, is treated as a one-column matrix.
predictor_weights	optional numeric vector of length ‘ J ’ giving ‘ $\text{colSums}(X^2)$ ’. Computed internally when ‘NULL’. Callers that have this cached on the data object pass it through to avoid recomputation.
sigma2	optional numeric scalar giving a known residual variance. When supplied, ‘ $\text{Shat}[j, t] = \sqrt{\text{sigma2} / \text{predictor_weights}[j]}$ ’ (single-effect-residual form). When ‘NULL’, ‘Shat’ is the per-pair empirical residual standard error: for each ‘ (j, t) ’ pair, the sample SD of ‘ $Y[, t] - X[, j] * \text{Bhat}[j, t]$ ’ divided by ‘ $\sqrt{n - 1}$ ’. The latter matches the form used by data-driven prior init routines (e.g., for fitting a normal-mixture prior via ‘ashr::ash’).

Value

list with elements ‘Bhat’ ($J \times T$) and ‘Shat’ ($J \times T$).

Examples

```

set.seed(1)
X <- matrix(rnorm(50 * 5), 50, 5)
X <- scale(X, center = TRUE, scale = FALSE)
Y <- matrix(rnorm(50 * 3), 50, 3)
out <- compute_marginal_bhat_shat(X, Y)
dim(out$Bhat)  # 5 x 3
dim(out$Shat)  # 5 x 3

```

compute_suff_stat	<i>Compute sufficient statistics for input to susie_ss</i>
-------------------	--

Description

Computes the sufficient statistics $X'X$, $X'y$, $y'y$ and n after centering (and possibly standardizing) the columns of X and centering y to have mean zero. We also store the column means of X and mean of y .

Usage

```
compute_suff_stat(X, y, standardize = FALSE)
```

Arguments

X	An n by p matrix of covariates.
y	An n vector.
standardize	Logical flag indicating whether to standardize columns of X to unit variance prior to computing summary data

Value

A list of sufficient statistics (XtX , Xty , yty , n) and $X_colmeans$, y_mean .

Examples

```

data(N2finemapping)
ss <- compute_suff_stat(N2finemapping$X, N2finemapping$Y[, 1])

```

data_small	<i>Simulated Small-sample eQTL Data.</i>
------------	--

Description

A simulated eQTL data set with 47 individuals and 7,430 variables. The response is a simulated gene expression phenotype and the variables are genotypes. This data set illustrates the small sample-size setting considered in Denault *et al* (2025).

Format

data_small is a list with the following elements:

y Simulated gene expression response.

X Genotype matrix.

References

W. R. P. Denault *et al* (2025). Accounting for uncertainty in residual variances improves fine-mapping in small sample studies. *bioRxiv* doi:10.1101/2025.05.16.654543.

See Also

The “Small data example” vignette.

Examples

```
data(data_small)
```

estimate_s_rss	<i>Estimate s in susie_rss Model Using Regularized LD</i>
----------------	---

Description

The estimated s gives information about the consistency between the z scores and LD matrix. A larger s means there is a strong inconsistency between z scores and LD matrix. The “null-mle” method obtains mle of s under $z|R \sim N(0, (1-s)R + sI)$, $0 < s < 1$. The “null-partialmle” method obtains mle of s under $U^T z|R \sim N(0, sI)$, in which U is a matrix containing the of eigenvectors that span the null space of R ; that is, the eigenvectors corresponding to zero eigenvalues of R . The estimated s from “null-partialmle” could be greater than 1. The “null-pseudomle” method obtains mle of s under pseudolikelihood $L(s) = \prod_{j=1}^p p(z_j|z_{-j}, s, R)$, $0 < s < 1$.

Usage

```
estimate_s_rss(z, R, n, r_tol = 1e-08, method = "null-mle")
```

Arguments

<code>z</code>	A p-vector of z scores.
<code>R</code>	A p by p symmetric, positive semidefinite correlation matrix.
<code>n</code>	The sample size. (Optional, but highly recommended.)
<code>r_tol</code>	Tolerance level for eigenvalue check of positive semidefinite matrix of R.
<code>method</code>	a string specifies the method to estimate s .

Value

A number between 0 and 1.

Examples

```
set.seed(1)
n <- 500
p <- 1000
beta <- rep(0, p)
beta[1:4] <- 0.01
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
X <- scale(X, center = TRUE, scale = TRUE)
y <- drop(X %%% beta + rnorm(n))
input_ss <- compute_suff_stat(X, y, standardize = TRUE)
ss <- univariate_regression(X, y)
R <- cor(X)
attr(R, "eigen") <- eigen(R, symmetric = TRUE)
zhat <- with(ss, betahat / sebetahat)

# Estimate s using the unadjusted z-scores.
s0 <- estimate_s_rss(zhat, R)

# Estimate s using the adjusted z-scores.
s1 <- estimate_s_rss(zhat, R, n)
```

 FinemappingConvergence

Simulated Fine-mapping Data with Convergence Problem.

Description

Data simulated using real genotypes from 50,000 individuals and 200 SNPs. Two of the SNPs have non-zero effects on the multivariate response. The response data are generated under a linear regression model. The simulated response and the columns of the genotype matrix are centered.

Format

FinemappingConvergence is a list with the following elements:

XtX Summary statistics computed using the centered and scaled genotype matrix.

Xty Summary statistics computed using the centered and scaled genotype data, and the centered simulated response.

yty yty is computed using the centered simulated response.

n The sample size (50,000).

true_coef The coefficients used to simulate the responses.

z z-scores from a simple (single-SNP) linear regression.

See Also

A similar data set with more SNPs is used in the “Refine SuSiE model” vignette.

Examples

```
data(FinemappingConvergence)
```

get.full.posterior *Approximation Posterior Expectations from Mr.ASH Fit*

Description

Recover the parameters specifying the variational approximation to the posterior distribution of the regression coefficients. To streamline the model fitting implementation, and to reduce memory requirements, `mr.ash` does not store all the parameters needed to specify the approximate posterior.

Usage

```
get.full.posterior(fit)
```

Arguments

`fit` A Mr.ASH fit obtained, for example, by running `mr.ash`.

Value

A list object with the following elements:

<code>phi</code>	An $p \times K$ matrix containing the posterior assignment probabilities, where p is the number of predictors, and K is the number of mixture components. (Each row of <code>phi</code> should sum to 1.)
<code>m</code>	An $p \times K$ matrix containing the posterior means conditional on assignment to each mixture component.
<code>s2</code>	An $p \times K$ matrix containing the posterior variances conditional on assignment to each mixture component.

Examples

```
## generate synthetic data
set.seed(1)
n      = 200
p      = 300
X      = matrix(rnorm(n*p),n,p)
beta   = double(p)
beta[1:10] = 1:10
y      = X %%% beta + rnorm(n)

## fit mr.ash
fit.mr.ash = mr.ash(X, y)

## recover full posterior
full.post  = get.full.posterior(fit.mr.ash)
```

get_cs_correlation	<i>Get Correlations Between CSs, using Variable with Maximum PIP From Each CS</i>
--------------------	---

Description

This function evaluates the correlation between single effect CSs. It is not part of the SuSiE inference. Rather, it is designed as a diagnostic tool to assess how correlated the reported CS are.

Usage

```
get_cs_correlation(model, X = NULL, Xcorr = NULL, max = FALSE)
```

Arguments

model	A SuSiE fit, typically an output from susie or one of its variants.
X	n by p matrix of values of the p variables (covariates) in n samples. When provided, correlation between variables will be computed and used to remove CSs whose minimum correlation among variables is smaller than <code>min_abs_corr</code> .
Xcorr	p by p matrix of correlations between variables (covariates). When provided, it will be used to remove CSs whose minimum correlation among variables is smaller than <code>min_abs_corr</code> .
max	When <code>max = FALSE</code> , return a matrix of CS correlations. When <code>max = TRUE</code> , return only the maximum absolute correlation among all pairs of correlations.

Value

A matrix of correlations between CSs, or the maximum absolute correlation when `max = TRUE`.

kriging_rss

*Compute Distribution of z-scores of Variant j Given Other z-scores,
and Detect Possible Allele Switch Issue*

Description

Under the null, the rss model with regularized LD matrix is $z_j | R, s \sim N(0, (1-s)R + sI)$. We use a mixture of normals to model the conditional distribution of z_j given other z scores, $z_j | z_{-j}, R, s \sim \sum_{k=1}^K \pi_k N(-\Omega_{j,-j} z_{-j} / \Omega_{jj}, \Omega_{jj})$. $\Omega = ((1-s)R + sI)^{-1}$, $\sigma_1, \dots, \sigma_K$ is a grid of fixed positive numbers. We estimate the mixture weights π . We detect the possible allele switch issue using likelihood ratio for each variant.

Usage

```
kriging_rss(
  z,
  R,
  n,
  r_tol = 1e-08,
  s = estimate_s_rss(z, R, n, r_tol, method = "null-mle")
)
```

Arguments

z	A p-vector of z scores.
R	A p by p symmetric, positive semidefinite correlation matrix.
n	The sample size. (Optional, but highly recommended.)
r_tol	Tolerance level for eigenvalue check of positive semidefinite matrix of R.
s	an estimated s from estimate_s_rss

Value

a list containing a ggplot2 plot object and a table. The plot compares observed z score vs the expected value. The possible allele switched variants are labeled as red points ($\log \text{LR} > 2$ and $\text{abs}(z) > 2$). The table summarizes the conditional distribution for each variant and the likelihood ratio test. The table has the following columns: the observed z scores, the conditional expectation, the conditional variance, the standardized differences between the observed z score and expected value, the log likelihood ratio statistics.

Examples

```
# See also the vignette, "Diagnostic for fine-mapping with summary
# statistics."
set.seed(1)
n <- 500
p <- 1000
beta <- rep(0, p)
beta[1:4] <- 0.01
```

```

X <- matrix(rnorm(n * p), nrow = n, ncol = p)
X <- scale(X, center = TRUE, scale = TRUE)
y <- drop(X %*% beta + rnorm(n))
ss <- univariate_regression(X, y)
R <- cor(X)
attr(R, "eigen") <- eigen(R, symmetric = TRUE)
zhat <- with(ss, betahat / sebetahat)
cond_dist <- kriging_rss(zhat, R, n = n)
cond_dist$plot

```

mr.ash

Multiple Regression with Adaptive Shrinkage

Description

Model fitting algorithms for Multiple Regression with Adaptive Shrinkage ("Mr.ASH"). Mr.ASH is a variational empirical Bayes (VEB) method for multiple linear regression. The fitting algorithms (locally) maximize the approximate marginal likelihood (the "evidence lower bound", or ELBO) via coordinate-wise updates.

Usage

```

mr.ash(
  X,
  y,
  Z = NULL,
  sa2 = NULL,
  method_q = c("sigma_dep_q", "sigma_indep_q"),
  method_g = c("caisa", "accelerate", "block"),
  max.iter = 1000,
  min.iter = 1,
  beta.init = NULL,
  update.pi = TRUE,
  pi = NULL,
  update.sigma2 = TRUE,
  sigma2 = NULL,
  update.order = NULL,
  standardize = FALSE,
  intercept = TRUE,
  tol = set_default_tolerance(),
  verbose = TRUE
)

```

Arguments

X The input matrix, of dimension (n,p); each column is a single predictor; and each row is an observation vector. Here, n is the number of samples and p is the number of predictors. The matrix cannot be sparse.

<code>y</code>	The observed continuously-valued responses, a vector of length p .
<code>Z</code>	The covariate matrix, of dimension (n,k) , where k is the number of covariates. This feature is not yet implemented; <code>Z</code> must be set to <code>NULL</code> .
<code>sa2</code>	The vector of prior mixture component variances. The variances should be in increasing order, starting at zero; that is, <code>sort(sa2)</code> should be the same as <code>sa2</code> . When <code>sa2</code> is <code>NULL</code> , the default setting is used, $sa2[k] = (2^{(0.05*(k-1))} - 1)^2$, for $k = 1:20$. For this default setting, $sa2[1] = 0$, and $sa2[20]$ is roughly 1.
<code>method_q</code>	The algorithm used to update the variational approximation to the posterior distribution of the regression coefficients, <code>method = "sigma_dep_q"</code> and <code>method = "sigma_indep_q"</code> , take different approaches to updating the residual variance σ^2 .
<code>method_g</code>	<code>method = "caisa"</code> , an abbreviation of "Coordinate Ascent Iterative Shrinkage Algorithm", fits the model by approximate EM; it iteratively updates the variational approximation to the posterior distribution of the regression coefficients (the approximate E-step) and the model parameters (mixture weights and residual covariance) in an approximate M-step. Settings <code>method = "block"</code> and <code>method = "accelerate"</code> are considered experimental.
<code>max.iter</code>	The maximum number of outer loop iterations allowed.
<code>min.iter</code>	The minimum number of outer loop iterations allowed.
<code>beta.init</code>	The initial estimate of the (approximate) posterior mean regression coefficients. This should be <code>NULL</code> , or a vector of length p . When <code>beta.init</code> is <code>NULL</code> , the posterior mean coefficients are all initially set to zero.
<code>update.pi</code>	If <code>update.pi = TRUE</code> , the mixture proportions in the mixture-of-normals prior are estimated from the data. In the manuscript, <code>update.pi = TRUE</code> .
<code>pi</code>	The initial estimate of the mixture proportions $\pi_1, \dots, \pi_K$. If <code>pi</code> is <code>NULL</code> , the mixture weights are initialized to <code>rep(1/K, K)</code>
, where $K = \text{length}(sa2)$.	
<code>update.sigma2</code>	If <code>update.sigma2 = TRUE</code> , the residual variance σ^2 is estimated from the data. In the manuscript, <code>update.sigma = TRUE</code> .
<code>sigma2</code>	The initial estimate of the residual variance, σ^2 . If <code>sigma2 = NULL</code> , the residual variance is initialized to the empirical variance of the residuals based on the initial estimates of the regression coefficients, <code>beta.init</code> , after removing linear effects of the intercept and any covariances.
<code>update.order</code>	The order in which the co-ordinate ascent updates for estimating the posterior mean coefficients are performed. <code>update.order</code> can be <code>NULL</code> , "random", or any permutation of $(1, \dots, p)$, where p is the number of columns in the input matrix X . When <code>update.order</code> is <code>NULL</code> , the co-ordinate ascent updates are performed in order in which they appear in X ; this is equivalent to setting <code>update.order = 1:p</code> . When <code>update.order = "random"</code> , the co-ordinate ascent updates are performed in a randomly generated order, and this random ordering is different at each outer-loop iteration.
<code>standardize</code>	The logical flag for standardization of the columns of X variable, prior to the model fitting. The coefficients are always returned on the original scale.

intercept	When <code>intercept = TRUE</code> , an intercept is included in the regression model.
tol	Additional settings controlling behaviour of the model fitting algorithm. <code>tol\$convtol</code> controls the termination criterion for the model fitting. The outer-loop updates stop when the relative L2 change in the estimates of the posterior mean coefficients is less than <code>convtol</code> , i.e., $ \text{beta_new} - \text{beta_old} _2 / \max(1, \text{beta_old} _2) < \text{convtol}$. <code>tol\$epstol</code> is a small, positive number added to the likelihoods to avoid logarithms of zero.
verbose	If <code>verbose = TRUE</code> , some information about the status of the model fitting is printed to the console.

Details

Mr.ASH is a statistical inference method for the following multiple linear regression model:

$$y|X, \beta, \sigma^2 \sim N(X\beta, \sigma^2 I_n),$$

in which the regression coefficients β admit a mixture-of-normals prior,

$$\beta|\pi, \sigma^2 \sim \sum_{k=1}^K N(0, \sigma_k^2).$$

Each mixture component in the prior, g , is a normal density centered at zero, with variance $\sigma^2 \sigma_k^2$.

The fitting algorithm, it run for a large enough number of iterations, will find an approximate posterior for the regression coefficients, denoted by $q(\beta)$, residual variance parameter σ^2 , and prior mixture weights $\pi_1, \dots, \pi_K$ maximizing the evidence lower bound,

$$F(q, \pi, \sigma^2) = E_q \log p(y|X, \beta, \sigma^2) - \sum_{j=1}^p D_{KL}(q_j || g),$$

where $D_{KL}(q_j || g)$ denotes the Kullback-Leibler (KL) divergence, a measure of the "distance" between (approximate) posterior $q_j(\beta_j)$ and prior $g(\beta_j)$. The fitting algorithm iteratively updates the approximate posteriors $q_1, \dots, q_p$, separately for each $j = 1, \dots, p$ (in an order determined by `update.order`), then separately updates the mixture weights and π and residual variance σ^2 . This coordinate-wise update scheme iterates until the convergence criterion is met, or until the algorithm hits an upper bound on the number of iterations (specified by `max.iter`). Coordinate-wise optimization algorithms for model fitting are implemented in C++ for efficient handling of large-scale data

See ‘References’ for more details about the model and algorithm.

Value

A list object with the following elements:

intercept	The estimated intercept.
beta	A vector containing posterior mean estimates of the regression coefficients for all predictors.
sigma2	The estimated residual variance.

<code>pi</code>	A vector of containing the estimated mixture proportions.
<code>iter</code>	The number of outer-loop iterations that were performed.
<code>update.order</code>	The ordering used for performing the coordinate-wise updates. For <code>update.order = "random"</code> , the orderings for outer-loop iterations are provided in a vector of length <code>p*max.iter</code> , where <code>p</code> is the number of predictors.
<code>varobj</code>	A vector of length <code>numiter</code> , containing the value of the variational objective (equal to the negative "evidence lower bound") attained at each (outer-loop) model fitting iteration. Note that the objective does not account for the intercept term, even when <code>intercept = TRUE</code> ; therefore, this value should be interpreted as being an approximation to the marginal likelihood <i>conditional</i> on the estimate of the intercept.
<code>data</code>	The preprocessed data (X, Z, y) provided as input to the model fitting algorithm. <code>data\$w</code> is equal to <code>diag(crossprod(X))</code> , in which X is the preprocessed data matrix. Additionally, <code>data\$sa2</code> gives the prior variances used.

References

Y. Kim (2020), Bayesian shrinkage methods for high dimensional regression. Ph.D. thesis, University of Chicago.

See Also

[get.full.posterior](#), [predict.mr.ash](#)

Examples

```
### generate synthetic data
set.seed(1)
n      = 200
p      = 300
X      = matrix(rnorm(n*p),n,p)
beta   = double(p)
beta[1:10] = 1:10
y      = X %%% beta + rnorm(n)

### fit Mr.ASH
fit.mr.ash = mr.ash(X,y, method_q = "sigma_indep_q")
#' fit.mr.ash = mr.ash(X,y, method_q = "sigma_dep_q")

### prediction routine
Xnew     = matrix(rnorm(n*p),n,p)
ynew     = Xnew %%% beta + rnorm(n)
ypred    = predict(fit.mr.ash, Xnew)

### test error
rmse     = norm(ynew - ypred, '2') / sqrt(n)

### coefficients
betahat  = predict(fit.mr.ash, type = "coefficients")
# this equals c(fit.mr.ash$intercept, fit.mr.ash$beta)
```

mr.ash.rss

Bayesian Multiple Regression with Mixture-of-Normals Prior (RSS)

Description

This function performs Bayesian multiple regression with a mixture-of-normals prior using summary statistics (RSS: Regression with Summary Statistics). It uses a C++ implementation for efficient computation.

Usage

```
mr.ash.rss(
  bhat,
  shat,
  R,
  var_y,
  n,
  s0,
  w0,
  sigma2_e = NULL,
  mu1_init = numeric(0),
  tol = 1e-08,
  max_iter = 1e+05,
  z = numeric(0),
  update_w0 = TRUE,
  update_sigma = TRUE,
  compute_ELBO = TRUE,
  standardize = FALSE
)
```

Arguments

bhat	Numeric vector of observed effect sizes (standardized).
shat	Numeric vector of standard errors of effect sizes.
R	Numeric matrix of the correlation matrix.
var_y	Numeric value of the variance of the outcome. If NULL, it is set to Inf (effects on standardized scale).
n	Integer value of the sample size.
s0	Numeric vector of prior variances for the mixture components.
w0	Numeric vector of prior weights for the mixture components.
sigma2_e	Numeric value of the initial error variance estimate. If NULL (default), initialized to var_y (matching mr.ash behavior of using residual variance with zero initialization), or 1 when var_y = Inf.

<code>mu1_init</code>	Numeric vector of initial values for the posterior mean of the coefficients. Default is <code>numeric(0)</code> (initialize to zero).
<code>tol</code>	Numeric value of the convergence tolerance. Default is <code>1e-8</code> .
<code>max_iter</code>	Integer value of the maximum number of iterations. Default is <code>1e5</code> .
<code>z</code>	Numeric vector of Z-scores. If not provided, computed as <code>bhat / shat</code> .
<code>update_w0</code>	Logical value indicating whether to update the mixture weights. Default is <code>TRUE</code> .
<code>update_sigma</code>	Logical value indicating whether to update the error variance. Default is <code>TRUE</code> .
<code>compute_ELBO</code>	Logical value indicating whether to compute the Evidence Lower Bound (ELBO). Default is <code>TRUE</code> .
<code>standardize</code>	Logical value indicating whether to standardize the input data. Default is <code>FALSE</code> .

Value

A list containing the following components:

beta Numeric vector of posterior mean coefficients (same as `mu1`).

sigma2 Numeric value of the residual variance (same as `sigma2_e`).

pi Numeric vector of mixture weights (same as `w0`).

iter Integer, number of iterations performed.

varobj Numeric vector of ELBO values per iteration.

mu1 Numeric vector of the posterior mean of the coefficients.

sigma2_1 Numeric vector of the posterior variance of the coefficients.

w1 Numeric matrix of the posterior assignment probabilities.

sigma2_e Numeric value of the error variance.

w0 Numeric vector of the mixture weights.

ELBO Numeric value of the Evidence Lower Bound (if `compute_ELBO = TRUE`).

Examples

```
# Generate example data
set.seed(985115)
n <- 350
p <- 16
sigmasq_error <- 0.5
zeroes <- rbinom(p, 1, 0.6)
beta.true <- rnorm(p, 1, sd = 4)
beta.true[zeroes] <- 0

X <- cbind(matrix(rnorm(n * p), nrow = n))
X <- scale(X, center = TRUE, scale = FALSE)
y <- X %*% matrix(beta.true, ncol = 1) + rnorm(n, 0, sqrt(sigmasq_error))
y <- scale(y, center = TRUE, scale = FALSE)

# Set the prior
```

```

K <- 9
sigma0 <- c(0.001, .1, .5, 1, 5, 10, 20, 30, .005)
omega0 <- rep(1 / K, K)

# Calculate summary statistics
b.hat <- sapply(1:p, function(j) {
  summary(lm(y ~ X[, j]))$coefficients[-1, 1]
})
s.hat <- sapply(1:p, function(j) {
  summary(lm(y ~ X[, j]))$coefficients[-1, 2]
})
R.hat <- cor(X)
var_y <- var(y)
sigmasq_init <- 1.5

# Run mr.ash.rss
out <- mr.ash.rss(b.hat, s.hat,
  R = R.hat, var_y = var_y, n = n,
  sigma2_e = sigmasq_init, s0 = sigma0, w0 = omega0,
  mul_init = rep(0, ncol(X)), tol = 1e-8, max_iter = 1e5,
  update_w0 = TRUE, update_sigma = TRUE, compute_ELBO = TRUE,
  standardize = FALSE
)

```

N2finemapping

Simulated Fine-mapping Data with Two Effect Variables

Description

This data set contains a genotype matrix for 574 individuals and 1,002 variables. The variables are genotypes after centering and scaling, and therefore retain the correlation structure of the original genotype data. Two of the variables have non-zero effects on the multivariate response. The response data are generated under a multivariate linear regression model. See Wang *et al* (2020) for details.

Format

N2finemapping is a list with the following elements:

X Centered and scaled genotype data.

chrom Chromosome of the original data, in hg38 coordinates.

pos Chromosomal position of the original data, in hg38 coordinates. The information can be used to compare impact of using other genotype references of the same variables in susie_rss application.

true_coef Simulated effect sizes.

residual_variance Simulated residual covariance matrix.

Y Simulated multivariate response.

allele_freq Allele frequencies based on the original genotype data.

V Suggested prior covariance matrix for effect sizes of the two non-zero effect variables.

References

G. Wang, A. Sarkar, P. Carbonetto and M. Stephens (2020). A simple new approach to variable selection in regression, with application to genetic fine-mapping. *Journal of the Royal Statistical Society, Series B* doi:10.1101/501114.

Examples

```
data(N2finemapping)
```

N3finemapping	<i>Simulated Fine-mapping Data with Three Effect Variables.</i>
---------------	---

Description

The data-set contains a matrix of 574 individuals and 1,001 variables. These variables are real-world genotypes centered and scaled, and therefore retains the correlation structure of variables in the original genotype data. 3 out of the variables have non-zero effects. The response data is generated under a multivariate linear regression model. See Wang *et al* (2020) for more details.

Format

N3finemapping is a list with the following elements:

X N by P variable matrix of centered and scaled genotype data.

chrom Chromosome of the original data, in hg38 coordinate.

pos Chromosomal position of the original data, in hg38 coordinate. The information can be used to compare impact of using other genotype references of the same variables in susie_rss application.

true_coef The simulated effect sizes.

residual_variance The simulated residual covariance matrix.

Y The simulated response variables.

allele_freq Allele frequency of the original genotype data.

V Prior covariance matrix for effect size of the three non-zero effect variables.

References

G. Wang, A. Sarkar, P. Carbonetto and M. Stephens (2020). A simple new approach to variable selection in regression, with application to genetic fine-mapping. *Journal of the Royal Statistical Society, Series B* doi:10.1101/501114.

Examples

```
data(N3finemapping)
```

path.order

*Ordering of Predictors by Regularization Path***Description**

This function determines an ordering of the predictors based on the regularization path of the penalized regression; in particular, the predictors are ordered based on the order in which the coefficients are included in the model as the penalty strength decreases.

Usage

```
path.order(fit)
```

Arguments

fit A fit object whose `coef()` method returns a matrix of coefficients with the intercept in the first row and one column per penalty strength (as produced by typical penalized-regression implementations).

Value

An ordering of the predictors.

Examples

```
### generate synthetic data
set.seed(1)
n      = 200
p      = 30
X      = matrix(rnorm(n*p),n,p)
beta   = double(p)
beta[1:10] = 1:10
y      = X %*% beta + rnorm(n)

### build a minimal example 'fit' object with the same structure as a
### fit from a penalized regression: a coefficient matrix with the
### intercept in row 1 and one column per (decreasing) penalty value.
beta_path = matrix(0, p + 1, p)
for (k in 1:p) beta_path[k + 1, k:p] = 1
fit       = list(coefficients = beta_path)
order     = path.order(fit)
```

predict.mr.ash

Predict Outcomes or Extract Coefficients from Mr.ASH Fit

Description

This function predicts outcomes (y) given the observed variables (X) and a Mr.ASH model; alternatively, retrieve the estimates of the regression coefficients.

Usage

```
## S3 method for class 'mr.ash'
predict(object, newx = NULL, type = c("response", "coefficients"), ...)
```

Arguments

object	A mr_ash fit, usually the result of calling mr.ash.
newx	The input matrix, of dimension (n,p); each column is a single predictor; and each row is an observation vector. Here, n is the number of samples and p is the number of predictors. When newx is NULL, the fitted values for the training data are provided.
type	The type of output. For type = "response", predicted or fitted outcomes are returned; for type = "coefficients", the estimated coefficients are returned.
...	Additional arguments passed to the default S3 method.

Value

For type = "response", predicted or fitted outcomes are returned; for type = "coefficients", the estimated coefficients are returned.

Examples

```
## generate synthetic data
set.seed(1)
n      = 200
p      = 300
X      = matrix(rnorm(n*p),n,p)
beta   = double(p)
beta[1:10] = 1:10
y      = X %%% beta + rnorm(n)

## fit mr.ash
fit.mr.ash = mr.ash(X, y)

## predict
Xnew      = matrix(rnorm(n*p),n,p)
ypred     = predict(fit.mr.ash, Xnew)
```

predict.susie	<i>Predict outcomes or extract coefficients from susie fit.</i>
---------------	---

Description

Predict outcomes or extract coefficients from susie fit.

Usage

```
## S3 method for class 'susie'
predict(object, newx = NULL, type = c("response", "coefficients"), ...)
```

Arguments

object	A susie fit.
newx	A new value for X at which to do predictions.
type	The type of output. For type = "response", predicted or fitted outcomes are returned; for type = "coefficients", the estimated coefficients are returned.
...	Other arguments used by generic predict function. These extra arguments are not used here.

Value

For type = "response", predicted or fitted outcomes are returned; for type = "coefficients", the estimated coefficients are returned. If the susie fit has intercept = NA (which is common when using susie_ss) then predictions are computed using an intercept of 0, and a warning is emitted.

print.summary.susie_post_outcome_configuration	<i>Print a summary.susie_post_outcome_configuration object</i>
--	--

Description

Pretty-prints the tidy tables built by [summary.susie_post_outcome_configuration()] with optional ANSI color highlighting. See that page for the color encoding.

Usage

```
## S3 method for class 'summary.susie_post_outcome_configuration'
print(x, ...)
```

Arguments

x	Output of [summary.susie_post_outcome_configuration()].
...	Ignored.

Value

The input invisibly.

See Also

[summary.susie_post_outcome_configuration()]

rss_mismatch_cases	<i>Simulated Summary Statistics with LD Reference Mismatch.</i>
--------------------	---

Description

Five simulated regions, each with one causal variant and 121 variants, used to illustrate how `susie_rss` handles a mismatch between the summary statistics and the LD reference. The first four cases simulate a phenotype from GTEx genotypes ($n = 838$); the last is a meta-analysis of an African-ancestry and a European-ancestry cohort.

Format

`rss_mismatch_cases` is a list of five cases. Each case is a list with the inputs to `susie_rss`:

z z-scores.

R LD matrix; for `multi_panel`, a named list of LD matrices, one per reference panel.

X For `ld_mismatch_sketch` only, a genotype sketch (factor matrix) of the reference panel, used instead of `R`. This case also stores `R`, a copy of `R_insample`.

n Sample size of the summary statistics.

B Reference panel sample size, or the sketch dimension (number of rows of `X`), passed as `R_finite`; one value per panel for `multi_panel`. Absent for `allele_flip`, whose LD is in-sample.

causal Index of the true causal variant.

The cases are:

allele_flip In-sample LD, with the sign of the z-score reversed at 1% of the variants. Also contains `z_clean`, the z-scores before flipping, and `flip_id`, the indices of the flipped variants.

ld_mismatch LD from 489 European individuals in 1000 Genomes.

ld_mismatch_sketch The reference panel given as a 500-row genotype sketch `X` of the ADSP panel.

ser_fallback LD from 489 European individuals in 1000 Genomes, a region where `SER_fallback` is TRUE.

multi_panel Meta-analysis of 400 African-ancestry and 400 European-ancestry samples, with AFR and EUR panel LD in `R`. Also contains `prop`, the true proportion of samples from each ancestry.

`ld_mismatch`, `ld_mismatch_sketch` and `ser_fallback` also contain `z_insample` and `R_insample`, the z-scores and the in-sample LD on the same variants.

See Also

The “Modeling and Accounting for LD Reference Mismatch in Summary Statistics Fine-mapping” vignette.

Examples

```
data(rss_mismatch_cases)
names(rss_mismatch_cases)
```

slot_prior_betabinom *Slot Activity Prior for SuSiE*

Description

Construct a prior specification for the slot activity model, which regularizes the number of active single effects in SuSiE. Two prior families are available: Beta-Binomial (default, recommended for single-locus) and Gamma-Poisson (recommended for genome-wide applications via susieAnn).

Usage

```
slot_prior_betabinom(
  a_beta = NULL,
  b_beta = NULL,
  c_hat_init = NULL,
  skip_threshold_multiplier = 0
)

slot_prior_poisson(
  C,
  nu = NULL,
  update_schedule = c("sequential", "batch"),
  c_hat_init = NULL,
  skip_threshold_multiplier = 0
)
```

Arguments

a_beta	Shape parameter for the Beta prior on inclusion probability rho. Default 1.
b_beta	Shape parameter for the Beta prior on inclusion probability rho. Default 2, giving a moderate sparsity preference with $E[\rho] = 1/3 \sim 0.33$. Setting a_beta = 1 and b_beta = 1 gives a uniform prior on [0,1], providing automatic multiplicity correction following Scott and Berger (2010).
c_hat_init	Optional numeric L-vector of initial slot activity probabilities for warm-starting. If NULL, initialized at the prior mean.

skip_threshold_multiplier	Multiplier for the adaptive skip threshold. Slots with $\hat{c}$ below this fraction of the baseline (prior with zero signal) are skipped. Default 0 (no skipping). The threshold is recomputed after each IBSS iteration from the current model state, and is set to 0 on the first iteration so all slots are evaluated at least once.
C	Expected number of causal variants for the Gamma-Poisson prior on the per-block causal rate. Must be positive. Not used by slot_prior_betabinom.
nu	Overdispersion parameter for the Gamma-Poisson prior on the per-block causal rate. Not used by slot_prior_betabinom. Larger values give stronger shrinkage toward C. Default 8 when not specified.
update_schedule	How the Gamma shape parameter is updated during IBSS iterations (Gamma-Poisson only; ignored for Beta-Binomial which is inherently sequential). "batch" updates once per full IBSS iteration (standard CAVI). "sequential" updates after each slot (faster convergence per iteration, used by susieAnn).

Details

Two prior types are available:

`slot_prior_betabinom` Uses a Beta-Binomial model for slot inclusion. The inclusion probability ρ is given a $\text{Beta}(a_{\text{beta}}, b_{\text{beta}})$ prior and integrated out analytically, yielding an adaptive multiplicity correction that penalizes less when more slots are active. This is the recommended default for single-locus applications. See Scott and Berger (2010) for the theoretical justification.

`slot_prior_poisson` Uses the Gamma-Poisson model with Poisson approximation for slot indicators. Recommended for genome-wide applications via `susieAnn`, where C and nu are estimated across loci.

Value

A list of class "slot_prior" with the appropriate subclass.

References

Scott, J. G. and Berger, J. O. (2010). Bayes and empirical-Bayes multiplicity adjustment in the variable-selection problem. *Annals of Statistics*, 38(5), 2587–2619.

Examples

```
# Default: Beta-Binomial with Beta(1, 2) prior on inclusion probability
slot_prior_betabinom()

# Gamma-Poisson for susieAnn
slot_prior_poisson(C = 5, nu = 8)

# Pass to susie
# fit <- susie(X, y, slot_prior = slot_prior_betabinom())
```

summary.susie	<i>Summarize Susie Fit.</i>
---------------	-----------------------------

Description

summary method for the “susie” class.

Usage

```
## S3 method for class 'susie'
summary(object, ...)

## S3 method for class 'summary.susie'
print(x, ...)
```

Arguments

object	A susie fit.
...	Additional arguments passed to the generic summary or print.summary method.
x	A susie summary.

Value

summary.susie returns a list containing a data frame of variables and a data frame of credible sets.

summary.susie_post_outcome_configuration	<i>Summarise a susie_post_outcome_configuration result</i>
--	--

Description

Builds tidy tables from the nested list returned by [susie_post_outcome_configuration()] and prints them with ANSI color highlighting via [print.summary.susie_post_outcome_configuration()]. The summary itself is an S3 object: index ‘\$susie_x’ and ‘\$coloc_pairwise’ to grab the data.frames for downstream use.

Usage

```
## S3 method for class 'susie_post_outcome_configuration'
summary(
  object,
  prob_thresh = 0.8,
  ambiguous_lower = 0.5,
  signal_only = TRUE,
  color = "auto",
  ...
)
```

Arguments

object	Output of [susie_post_outcome_configuration()].
prob_thresh	Threshold above which ‘marginal_prob’ counts as a signal (default ‘0.8’).
ambiguous_lower	Lower edge of the "ambiguous" band for the SuSiEx color coding: marginals in [‘ambiguous_lower, prob_thresh’) are colored yellow. Default ‘0.5’. Set to ‘prob_thresh’ to disable the band.
signal_only	Logical. If ‘TRUE’ (default), drop CS tuples where no trait is active and drop coloc rows whose dominant hypothesis is H0. Pass ‘FALSE’ to keep everything.
color	One of ‘"auto"’ (default; honors [crayon::has_color()]), ‘TRUE’ (force colors on), or ‘FALSE’ (force them off).
...	Ignored.

Details

Color encoding (when ANSI colors are available):

- SuSiEx per-trait marginal probability: bold dark green when ‘>= prob_thresh’ (active), yellow when in [‘ambiguous_lower, prob_thresh’), dim otherwise. The ‘active’ logical from the raw result is encoded by color and is not shown as a separate column.
- Coloc verdict: bold dark green for H4 (shared causal), magenta for H3 (distinct causals), blue for H1 or H2 (single-trait causal), dim for H0 (no signal). The dominant PP per row is bolded.

Robustness: this method is defensive against malformed input. Empty lists, NULL components, missing fields, length-mismatched per-trait vectors, trait names that collide with reserved columns (‘tuple’, ‘top_pattern’, ‘top_prob’), and coloc data.frames that lack some optional columns (‘hit1’, ‘hit2’) all degrade gracefully rather than erroring.

Value

A list of class ‘"summary.susie_post_outcome_configuration"' with components:

- **‘\$susie_x’** ‘data.frame’ (or ‘NULL’ when no signals): one row per CS tuple. Columns: ‘tuple’ (e.g. ‘“(1,1,1)”’), one numeric column per trait carrying that trait’s ‘marginal_prob’, ‘top_pattern’ (binary configuration string for the most-probable configuration), ‘top_prob’ (its probability).
- **‘\$coloc_pairwise’** ‘data.frame’ (or ‘NULL’): the original coloc table extended with ‘verdict’ (named hypothesis label) and ‘top_pp’ (the dominant PP value).
- **‘\$susie_x_n_total’, ‘\$susie_x_n_kept’, ‘\$coloc_n_total’, ‘\$coloc_n_kept’** row counts before and after ‘signal_only’ filtering, used by the print method to footer hidden rows.
- **‘\$prob_thresh’, ‘\$ambiguous_lower’, ‘\$signal_only’, ‘\$color’** parameters echoed for the print method.

See Also

[susie_post_outcome_configuration()], [print.summary.susie_post_outcome_configuration()]

SummaryConsistency	<i>Simulated Fine-mapping Data with LD matrix From Reference Panel.</i>
--------------------	---

Description

Data simulated using real genotypes from 10,000 individuals and 200 SNPs. One SNP have non-zero effect on the multivariate response. The response data are generated under a linear regression model. There is also one SNP with flipped allele between summary statistics and the reference panel.

Format

SummaryConsistency is a list with the following elements:

z z-scores computed by fitting univariate simple regression variable-by-variable.

ldref LD matrix estimated from the reference panel.

flip_id The index of the SNP with the flipped allele.

signal_id The index of the SNP with the non-zero effect.

See Also

A similar data set with more samples is used in the “Diagnostic for fine-mapping with summary statistics” vignette.

Examples

```
data(SummaryConsistency)
```

susie	<i>Sum of Single Effects (SuSiE) Regression</i>
-------	---

Description

Performs a sparse Bayesian multiple linear regression of y on X , using the "Sum of Single Effects" model from Wang et al (2020). In brief, this function fits the regression model $y = \mu + Xb + e$, where elements of e are *i.i.d.* normal with zero mean and variance `residual_variance`, μ is an intercept term and b is a vector of length p representing the effects to be estimated. The “susie assumption” is that $b = \sum_{l=1}^L b_l$ where each b_l is a vector of length p with exactly one non-zero element. The prior on the non-zero element is normal with zero mean and variance $\text{var}(y) * \text{scaled_prior_variance}$. The value of L is fixed, and should be chosen to provide a reasonable upper bound on the number of non-zero effects to be detected. Typically, the hyperparameters `residual_variance` and `scaled_prior_variance` will be estimated during model fitting, although they can also be fixed as specified by the user. See functions [susie_get_cs](#) and other functions of form `susie_get_*` to extract the most commonly-used results from a susie fit.

#' @details The function `susie` implements the IBSS algorithm from Wang et al (2020). The option `refine = TRUE` implements an additional step to help reduce problems caused by convergence of the IBSS algorithm to poor local optima (which is rare in our experience, but can provide misleading results when it occurs). The refinement step incurs additional computational expense that increases with the number of CSs found in the initial run.

The function `susie_ss` implements essentially the same algorithms, but using sufficient statistics. (The statistics are sufficient for the regression coefficients b , but not for the intercept μ ; see below for how the intercept is treated.) If the sufficient statistics are computed correctly then the results from `susie_ss` should be the same as (or very similar to) `susie`, although runtimes will differ as discussed below. The sufficient statistics are the sample size n , and then the p by p matrix $X'X$, the p -vector $X'y$, and the sum of squared y values $y'y$, all computed after centering the columns of X and the vector y to have mean 0; these can be computed using `compute_suff_stat`.

The handling of the intercept term in `susie_ss` needs some additional explanation. Computing the summary data after centering X and y effectively ensures that the resulting posterior quantities for b allow for an intercept in the model; however, the actual value of the intercept cannot be estimated from these centered data. To estimate the intercept term the user must also provide the column means of X and the mean of y (`X_colmeans` and `y_mean`). If these are not provided, they are treated as NA, which results in the intercept being NA. If for some reason you prefer to have the intercept be 0 instead of NA then set `X_colmeans = 0, y_mean = 0`.

For completeness, we note that if `susie_ss` is run on $X'X, X'y, y'y$ computed *without* centering X and y , and with `X_colmeans = 0, y_mean = 0`, this is equivalent to `susie` applied to X, y with `intercept = FALSE` (although results may differ due to different initializations of `residual_variance` and `scaled_prior_variance`). However, this usage is not recommended for most situations.

The computational complexity of `susie` is $O(npL)$ per iteration, whereas `susie_ss` is $O(p^2L)$ per iteration (not including the cost of computing the sufficient statistics, which is dominated by the $O(np^2)$ cost of computing $X'X$). Because of the cost of computing $X'X$, `susie` will usually be faster. However, if $n \gg p$, and/or if $X'X$ is already computed, then `susie_ss` may be faster.

Usage

```
susie(
  X,
  y,
  L = min(10, ncol(X)),
  scaled_prior_variance = 0.2,
  residual_variance = NULL,
  prior_weights = NULL,
  null_weight = 0,
  standardize = TRUE,
  intercept = TRUE,
  estimate_residual_variance = TRUE,
  estimate_residual_method = c("MoM", "MLE", "NIG"),
  estimate_prior_variance = TRUE,
  estimate_prior_method = c("optim", "EM", "simple"),
  prior_variance_grid = NULL,
  mixture_weights = NULL,
  unmapable_effects = c("none", "inf", "ash", "ash_filter_archived"),
```

```

check_null_threshold = 0,
prior_tol = 1e-09,
residual_variance_upperbound = Inf,
model_init = NULL,
s_init = NULL,
coverage = 0.95,
min_abs_corr = 0.5,
compute_univariate_zscore = FALSE,
na.rm = FALSE,
max_iter = 100,
L_greedy = NULL,
greedy_lbf_cutoff = 0.1,
tol = NULL,
convergence_method = c("elbo", "pip"),
verbose = FALSE,
track_fit = FALSE,
residual_variance_lowerbound = NULL,
refine = FALSE,
n_purity = "auto",
median_abs_corr = NULL,
cs_extension_corr = NULL,
alpha0 = NULL,
beta0 = NULL,
init_only = FALSE,
slot_prior = NULL
)

```

Arguments

<code>X</code>	An n by p matrix of covariates.
<code>y</code>	The observed responses, a vector of length n .
<code>L</code>	Maximum number of non-zero effects in the model. If L is larger than the number of covariates, p , L is set to p .
<code>scaled_prior_variance</code>	The prior variance, divided by $\text{var}(y)$ (or by $(1/(n-1))y^T y$ for <code>susie_ss</code>); that is, the prior variance of each non-zero element of b is $\text{var}(y) * \text{scaled_prior_variance}$. The value provided should be either a scalar or a vector of length L . If <code>estimate_prior_variance = TRUE</code> , this provides initial estimates of the prior variances.
<code>residual_variance</code>	Variance of the residual. If <code>estimate_residual_variance = TRUE</code> , this value provides the initial estimate of the residual variance. By default, it is set to $\text{var}(y)$ in <code>susie</code> and $(1/(n-1))y^T y$ in <code>susie_ss</code> .
<code>prior_weights</code>	A vector of length p , in which each entry gives the prior probability that corresponding column of X has a nonzero effect on the outcome, y . The weights are internally normalized to sum to 1. When <code>NULL</code> (the default), uniform prior weights are used (each variable is assigned probability $1/p$).
<code>null_weight</code>	Prior probability of no effect (a number between 0 and 1, and cannot be exactly 1).

standardize	If standardize = TRUE, standardize the columns of X to unit variance prior to fitting (or equivalently standardize XtX and Xty to have the same effect). Note that scaled_prior_variance specifies the prior on the coefficients of X <i>after</i> standardization (if it is performed). If you do not standardize, you may need to think more carefully about specifying scaled_prior_variance. Whatever your choice, the coefficients returned by coef are given for X on the original input scale. Any column of X that has zero variance is not standardized.
intercept	If intercept = TRUE, the intercept is fitted; if intercept = FALSE, the intercept is set to zero. Setting intercept = FALSE is generally not recommended.
estimate_residual_variance	If estimate_residual_variance = TRUE, the residual variance is estimated, using residual_variance as an initial value. If estimate_residual_variance = FALSE, the residual variance is fixed to the value supplied by residual_variance.
estimate_residual_method	The method used for estimating residual variance. For the original SuSiE model, "MLE" and "MoM" estimation is equivalent, but for the infinitesimal model, "MoM" is more stable. We recommend using "NIG" when $n < 80$ for improved coverage, although it is currently only implemented for individual-level data.
estimate_prior_variance	If estimate_prior_variance = TRUE, the prior variance is estimated (this is a separate parameter for each of the L effects). If provided, scaled_prior_variance is then used as an initial value for the optimization. When estimate_prior_variance = FALSE, the prior variance for each of the L effects is determined by the value supplied to scaled_prior_variance.
estimate_prior_method	The method used for estimating prior variance. When estimate_prior_method = "simple" is used, the likelihood at the specified prior variance is compared to the likelihood at a variance of zero, and the setting with the larger likelihood is retained. When prior_variance_grid is provided, this is automatically set to "fixed_mixture".
prior_variance_grid	Numeric vector of K prior variances defining a mixture-of-normals prior on effect sizes. When provided, the SER evaluates Bayes factors at each grid point and forms a mixture BF weighted by mixture_weights. This bypasses the scalar prior variance optimization. Default is NULL (standard scalar V path).
mixture_weights	Numeric vector of K non-negative weights summing to 1, giving the mixture proportions for the variance grid. Default is NULL, which uses uniform weights when prior_variance_grid is provided.
unmappable_effects	The method for modeling unmappable effects: "none", "inf", "ash".
check_null_threshold	When the prior variance is estimated, compare the estimate with the null, and set the prior variance to zero unless the log-likelihood using the estimate is larger by this threshold amount. For example, if you set check_null_threshold = 0.1, this will "nudge" the estimate towards zero when the difference in log-likelihoods is small. A note of caution that setting this to a value greater than

	zero may lead the IBSS fitting procedure to occasionally decrease the ELBO. This setting is disabled when using <code>unmappable_effects = "inf"</code> or <code>unmappable_effects = "ash"</code> .
<code>prior_tol</code>	When the prior variance is estimated, compare the estimated value to <code>prior_tol</code> at the end of the computation, and exclude a single effect from PIP computation if the estimated prior variance is smaller than this tolerance value.
<code>residual_variance_upperbound</code>	Upper limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
<code>model_init</code>	A previous susie fit with which to initialize.
<code>s_init</code>	Deprecated alias for <code>model_init</code> .
<code>coverage</code>	A number between 0 and 1 specifying the “coverage” of the estimated confidence sets.
<code>min_abs_corr</code>	Minimum absolute correlation allowed in a credible set. The default, 0.5, corresponds to a squared correlation of 0.25, which is a commonly used threshold for genotype data in genetic studies. This “purity” filter is applied to the CSs reported in the fit object, so the CS list returned here may be a subset of the one produced by calling <code>susie_get_cs</code> on the same fit without passing <code>X</code> or <code>Xcorr</code> (in which case the purity filter is skipped). Set to <code>NULL</code> to disable the minimum-correlation clause (e.g. to filter on <code>median_abs_corr</code> alone).
<code>compute_univariate_zscore</code>	If <code>compute_univariate_zscore = TRUE</code> , the univariate regression z-scores are outputted for each variable.
<code>na.rm</code>	Drop any missing values in <code>y</code> from both <code>X</code> and <code>y</code> .
<code>max_iter</code>	Maximum number of IBSS iterations to perform. For <code>susie_rss()</code> and <code>susie_rss_lambda()</code> , <code>NULL</code> uses 50 with a hint; other interfaces use 100.
<code>L_greedy</code>	Integer or <code>NULL</code> . When non- <code>NULL</code> , run a greedy outer loop that grows the number of effects from <code>L_greedy</code> up to <code>L</code> in linear steps until the fit saturates. The default <code>NULL</code> runs the usual fixed- <code>L</code> fit.
<code>greedy_lbf_cutoff</code>	Numeric saturation threshold for the <code>L_greedy</code> outer loop. Default is 0.1.
<code>tol</code>	A small, non-negative number specifying the convergence tolerance for the IBSS fitting procedure. When <code>NULL</code> , the default is $1e-4$; for <code>estimate_residual_method = "NIG"</code> , the default is tightened to $1e-6$.
<code>convergence_method</code>	When <code>convergence_method = "elbo"</code> the fitting procedure halts when the difference in the variational lower bound, or “ELBO” (the objective function to be maximized), is less than <code>tol</code> . When <code>convergence_method = "pip"</code> the fitting procedure halts when the maximum absolute difference in <code>alpha</code> is less than <code>tol</code> .
<code>verbose</code>	If <code>verbose = TRUE</code> , the algorithm’s progress, a summary of the optimization settings, and refinement progress (if <code>refine = TRUE</code>) are printed to the console.
<code>track_fit</code>	If <code>track_fit = TRUE</code> , a compact <code>susie_track</code> object is returned in <code>trace</code> , containing <code>alpha</code> history, effect summaries and available diagnostics at each iteration of the IBSS fitting procedure.

<code>residual_variance_lowerbound</code>	Lower limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
<code>refine</code>	If <code>refine = TRUE</code> , then an additional iterative refinement procedure is used, after the IBSS algorithm, to check and escape from local optima (see details).
<code>n_purity</code>	Passed as argument <code>n_purity</code> to susie_get_cs .
<code>median_abs_corr</code>	An optional second purity threshold applied to the median absolute correlation among the CS variables (default NULL, off). When both <code>min_abs_corr</code> and <code>median_abs_corr</code> are set, they are OR-linked: a CS is kept if it clears either one. See susie_get_cs .
<code>cs_extension_corr</code>	Passed as argument <code>cs_extension_corr</code> to susie_get_cs when the credible sets are constructed. Either NULL (the default, meaning no correlation-based extension) or a single number between 0 and 1. When set, each credible set is extended to absorb every variable whose absolute correlation with a set member exceeds the threshold (near-perfectly correlated proxies); 0.99 is the recommended value.
<code>alpha0</code>	Numerical parameter for the NIG prior when using <code>estimate_residual_method = "NIG"</code> . Defaults to $1/\sqrt{n}$, where n is the sample size. When calling <code>susie_rss</code> with NIG, n must be supplied; otherwise validation errors.
<code>beta0</code>	Numerical parameter for the NIG prior when using <code>estimate_residual_method = "NIG"</code> . Defaults to $1/\sqrt{n}$, where n is the sample size. When calling <code>susie_rss</code> with NIG, n must be supplied; otherwise validation errors.
<code>init_only</code>	Logical. If TRUE, return a list with data and params objects without running the IBSS algorithm. Used by packages like <code>susieAnn</code> that implement their own outer loop around SuSiE's building blocks. Default is FALSE.
<code>slot_prior</code>	Optional slot activity prior created by slot_prior_betabinom or slot_prior_poisson . Use <code>slot_prior_betabinom(a_beta, b_beta)</code> for the usual single-locus setting; it places a Beta-Binomial prior on the number of active effects and gives an adaptive multiplicity correction. Use <code>slot_prior_poisson(C, nu)</code> when you want a Gamma-Poisson prior centered on an expected number C of active effects. When supplied, each single-effect slot has an estimated activity probability c_{hat} ; fitted values and PIPs are weighted by these activity probabilities, and convergence is checked using <code>convergence_method = "pip"</code> .

Value

A "susie" object with some or all of the following elements:

<code>alpha</code>	An L by p matrix of posterior inclusion probabilities.
<code>mu</code>	An L by p matrix of posterior means, conditional on inclusion.
<code>mu2</code>	An L by p matrix of posterior second moments, conditional on inclusion.
<code>Xr</code>	A vector of length n , equal to $X \%*\% \text{colSums}(\alpha * \mu)$.
<code>lbf</code>	Log-Bayes Factor for each single effect.

lbf_variable	Log-Bayes Factor for each variable and single effect.
intercept	Intercept (fixed or estimated).
sigma2	Residual variance (fixed or estimated).
V	Prior variance of the non-zero elements of b.
elbo	The variational lower bound (or ELBO) achieved at each iteration.
fitted	Vector of length n containing the fitted values.
sets	Credible sets estimated from model fit.
pip	A vector of length p giving the marginal posterior inclusion probabilities.
z	A vector of univariate z-scores.
niter	Number of IBSS iterations performed.
converged	TRUE or FALSE indicating whether the IBSS converged to a solution within the chosen tolerance level.
theta	If unmappable_effects = "inf" or unmappable_effects = "ash", then theta is a p-vector of posterior means for the unmappable effects.
tau2	If unmappable_effects = "inf" or unmappable_effects = "ash", then tau2 is the unmappable variance.

susie_auto

Attempt at Automating SuSiE for Hard Problems

Description

susie_auto is an attempt to automate reliable running of susie even on hard problems. It implements a three-stage strategy for each L: first, fit susie with very small residual error; next, estimate residual error; finally, estimate the prior variance. If the last step estimates some prior variances to be zero, stop. Otherwise, double L, and repeat. Initial runs are performed with relaxed tolerance; the final run is performed using the default susie tolerance.

Usage

```
susie_auto(
  X,
  y,
  L_init = 1,
  L_max = 512,
  verbose = FALSE,
  init_tol = 1,
  standardize = TRUE,
  intercept = TRUE,
  max_iter = 100,
  tol = 0.01,
  ...
)
```

Arguments

<code>X</code>	An n by p matrix of covariates.
<code>y</code>	The observed responses, a vector of length n .
<code>L_init</code>	The initial value of L .
<code>L_max</code>	The largest value of L to consider.
<code>verbose</code>	If <code>verbose = TRUE</code> , the algorithm's progress, and a summary of the optimization settings, are printed to the console.
<code>init_tol</code>	The tolerance to passed to <code>susie</code> during early runs (set large to shorten the initial runs).
<code>standardize</code>	If <code>standardize = TRUE</code> , standardize the columns of X to unit variance prior to fitting. Note that <code>scaled_prior_variance</code> specifies the prior on the coefficients of X <i>after</i> standardization (if it is performed). If you do not standardize, you may need to think more carefully about specifying <code>scaled_prior_variance</code> . Whatever your choice, the coefficients returned by <code>coef</code> are given for X on the original input scale. Any column of X that has zero variance is not standardized.
<code>intercept</code>	If <code>intercept = TRUE</code> , the intercept is fitted; if <code>intercept = FALSE</code> , the intercept is set to zero. Setting <code>intercept = FALSE</code> is generally not recommended.
<code>max_iter</code>	Maximum number of IBSS iterations to perform.
<code>tol</code>	A small, non-negative number specifying the convergence tolerance for the IBSS fitting procedure. The fitting procedure will halt when the difference in the variational lower bound, or "ELBO" (the objective function to be maximized), is less than <code>tol</code> .
<code>...</code>	Additional arguments passed to susie .

Value

See [susie](#) for a description of return values.

See Also

[susie](#)

Examples

```
set.seed(1)
n = 1000
p = 1000
beta = rep(0,p)
beta[1:4] = 1
X = matrix(rnorm(n*p),nrow = n,ncol = p)
X = scale(X,center = TRUE,scale = TRUE)
y = drop(X %*% beta + rnorm(n))
res = susie_auto(X,y)
plot(beta,coef(res)[-1])
abline(a = 0,b = 1,col = "skyblue",lty = "dashed")
plot(y,predict(res))
abline(a = 0,b = 1,col = "skyblue",lty = "dashed")
```

Description

These functions access basic properties or draw inferences from a fitted susie model.

`susie_get_cs_attainable` returns credible sets after a post-hoc filter based on *attainable coverage* and per-effect entropy. Use this as a fallback when no LD reference is available for the standard purity filter in `susie_get_cs`.

Usage

```
susie_get_objective(res, last_only = TRUE, warning_tol = 1e-06)
```

```
susie_get_posterior_mean(res, prior_tol = 1e-09)
```

```
susie_get_posterior_sd(res, prior_tol = 1e-09)
```

```
susie_get_niter(res)
```

```
susie_get_prior_variance(res)
```

```
susie_get_residual_variance(res)
```

```
susie_get_lfsr(res)
```

```
susie_get_posterior_samples(susie_fit, num_samples)
```

```
susie_get_cs(  
  res,  
  X = NULL,  
  Xcorr = NULL,  
  coverage = 0.95,  
  min_abs_corr = 0.5,  
  dedup = TRUE,  
  squared = FALSE,  
  check_symmetric = TRUE,  
  n_purity = "auto",  
  median_abs_corr = NULL,  
  cs_extension_corr = NULL  
)
```

```
susie_get_cs_attainable(res, coverage = 0.95, ethres = NULL, ...)
```

```
susie_get_pip(res, prune_by_cs = FALSE, prior_tol = 1e-09)
```

Arguments

res	A susie fit, typically an output from susie or one of its variants. For <code>susie_get_pip</code> and <code>susie_get_cs</code> , this may instead be the posterior inclusion probability matrix, <code>alpha</code> .
last_only	If <code>last_only = FALSE</code> , return the ELBO from all iterations; otherwise return the ELBO from the last iteration only.
warning_tol	Warn if ELBO is decreasing by this tolerance level.
prior_tol	Filter out effects having estimated prior variance smaller than this threshold.
susie_fit	A susie fit, an output from susie .
num_samples	The number of draws from the posterior distribution.
X	n by p matrix of values of the p variables (covariates) in n samples. When provided, correlation between variables will be computed and used to remove CSs whose minimum correlation among variables is smaller than <code>min_abs_corr</code> .
Xcorr	p by p matrix of correlations between variables (covariates). When provided, it will be used to remove CSs whose minimum correlation among variables is smaller than <code>min_abs_corr</code> .
coverage	A number between 0 and 1 specifying desired coverage of each CS.
min_abs_corr	A "purity" threshold for the CS, applied to the minimum absolute correlation among the CS variables: a CS is filtered out unless this minimum is at least <code>min_abs_corr</code> . Set to <code>NULL</code> to disable this clause. This filter is only applied when X or Xcorr is provided; otherwise it is ignored and a warning is issued.
dedup	If <code>dedup = TRUE</code> , remove duplicate CSs.
squared	If <code>squared = TRUE</code> , report min, mean and median of squared correlation instead of the absolute correlation.
check_symmetric	If <code>check_symmetric = TRUE</code> , perform a check for symmetry of matrix Xcorr when Xcorr is provided (not <code>NULL</code>).
n_purity	Maximum number of CS variables used for purity when correlations are computed from X. The default, "auto", uses a resource-aware cap; a positive number gives a fixed cap; a negative number uses all CS variables. Xcorr inputs always use all variables.
median_abs_corr	An optional second purity threshold applied to the <i>median</i> absolute correlation among the CS variables. The default, <code>NULL</code> , leaves it off. When both <code>min_abs_corr</code> and <code>median_abs_corr</code> are non- <code>NULL</code> , they are combined with OR: a CS is kept if it clears <i>either</i> threshold (its $\min \geq \text{min_abs_corr}$ or its $\text{median} \geq \text{median_abs_corr}$).
cs_extension_corr	Either <code>NULL</code> or a single number between 0 and 1. If non- <code>NULL</code> , each credible set is extended to include every variable whose absolute correlation with a credible-set member exceeds this threshold, pulling in near-perfectly correlated proxies (variables statistically indistinguishable from the selected ones). Works from either X or Xcorr. The default, <code>NULL</code> , disables correlation-based extension; 0.99 is the recommended value when extension is desired.

ethres	Entropy threshold expressed as an effective number of variables: an effect is dropped when its attainable-projection entropy is at least $\log(\text{ethres})$. Defaults to $\max(100, 0.1 * p)$ where p is the number of variables.
...	Additional arguments (e.g. X or Xcorr) accepted for compatibility with <code>susie_get_cs</code> ; they are ignored here.
prune_by_cs	Whether or not to ignore single effects not in a reported CS when calculating PIP.

Details

For each effect l , the *attainable coverage* is computed by zeroing every entry of `res$alpha` that is not the column maximum, then summing across variables. Intuitively, it is the coverage effect l could attain if it did not have to share probability mass with other effects. Effects with attainable coverage at most coverage, or with attainable-projection entropy at least $\log(\text{ethres})$, are dropped before delegating to `susie_get_cs`. Any X or Xcorr passed through ... is ignored, since the procedure is intended for the no-LD setting.

Value

`susie_get_objective` returns the evidence lower bound (ELBO) achieved by the fitted `susie` model and, optionally, at each iteration of the IBSS fitting procedure.

`susie_get_residual_variance` returns the (estimated or fixed) residual variance parameter.

`susie_get_prior_variance` returns the (estimated or fixed) prior variance parameters.

`susie_get_posterior_mean` returns the posterior mean for the regression coefficients of the fitted `susie` model. Note this is not interchangeable with `coef`: `coef` sums over all L single-effect rows and prepends an intercept, whereas `susie_get_posterior_mean` first drops effects with estimated prior variance $\leq \text{prior_tol}$. The two agree on a converged fit and can differ otherwise.

`susie_get_posterior_sd` returns the posterior standard deviation for coefficients of the fitted `susie` model.

`susie_get_niter` returns the number of model fitting iterations performed.

`susie_get_pip` returns a vector containing the posterior inclusion probabilities (PIPs) for all variables.

`susie_get_lfsr` returns a vector containing the average lfsr across variables for each single-effect, weighted by the posterior inclusion probability (α).

`susie_get_posterior_samples` returns a list containing the effect sizes samples and causal status with two components: `b`, an `num_variables x num_samples` matrix of effect sizes; `gamma`, an `num_variables x num_samples` matrix of causal status random draws.

`susie_get_cs` returns credible sets (CSs) from a `susie` fit, as well as summaries of correlation among the variables included in each CS. If desired, one can filter out CSs that do not meet a specified “purity” threshold; to do this, either X or Xcorr must be supplied. It returns a list with the following elements:

cs	A list in which each list element is a vector containing the indices of the variables in the CS.
coverage	The nominal coverage specified for each CS.

purity	If X or Xcorr is provided), the purity of each CS.
cs_index	If X or Xcorr is provided) the index (number between 1 and L) of each reported CS in the supplied susie fit.

Examples

```

set.seed(1)
n <- 1000
p <- 1000
beta <- rep(0, p)
beta[1:4] <- 1
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
X <- scale(X, center = TRUE, scale = TRUE)
y <- drop(X %*% beta + rnorm(n))
s <- susie(X, y, L = 10)
susie_get_objective(s)
susie_get_objective(s, last_only = FALSE)
susie_get_residual_variance(s)
susie_get_prior_variance(s)
susie_get_posterior_mean(s)
susie_get_posterior_sd(s)
susie_get_niter(s)
susie_get_pip(s)
susie_get_lfsr(s)

```

susie_init_coef	<i>Initialize a susie object using regression coefficients</i>
-----------------	--

Description

Initialize a susie object using regression coefficients

Usage

```
susie_init_coef(coef_index, coef_value, p)
```

Arguments

coef_index	An L-vector containing the the indices of the nonzero coefficients.
coef_value	An L-vector containing initial coefficient estimates.
p	A scalar giving the number of variables.

Value

A list with elements alpha, mu and mu2 to be used by susie.

Examples

```
set.seed(1)
n = 1000
p = 1000
beta = rep(0,p)
beta[sample(1:1000,4)] = 1
X = matrix(rnorm(n*p),nrow = n,ncol = p)
X = scale(X,center = TRUE,scale = TRUE)
y = drop(X %*% beta + rnorm(n))

# Initialize susie to ground-truth coefficients.
s = susie_init_coef(which(beta != 0),beta[beta != 0],length(beta))
res = susie(X,y,L = 10,model_init=s)
```

susie_plot

SuSiE Plots.

Description

susie_plot produces a per-variable summary of the SuSiE credible sets. susie_plot_iteration produces a diagnostic plot for the susie model fitting. For susie_plot_iteration, several plots will be created if track_fit = TRUE when calling susie.

Usage

```
susie_plot(
  model,
  y,
  add_bar = FALSE,
  pos = NULL,
  b = NULL,
  max_cs = 400,
  add_legend = NULL,
  ...
)

susie_plot_iteration(model, L, file_prefix, pos = NULL)
```

Arguments

model	A SuSiE fit, typically an output from susie or one of its variants. For suse_plot, the susie fit must have model\$z, model\$PIP, and may include model\$sets. model may also be a vector of z-scores or PIPs. For susie_plot_iteration, model must be either a SuSiE fit or its susie_track object from model\$trace.
-------	---

y	A string indicating what to plot: either "z_original" for z-scores, "z" for z-score derived p-values on (base-10) log-scale, "PIP" for posterior inclusion probabilities, "log10PIP" for posterior inclusion probabilities on the (base-10) log-scale. For any other setting, the data are plotted as is.
add_bar	If add_bar = TRUE, add horizontal bar to signals in credible interval.
pos	Indices of variables to plot. If pos = NULL all variables are plotted.
b	For simulated data, set b = TRUE to highlight "true" effects (highlights in red).
max_cs	The largest credible set to display, either based on purity (set max_cs between 0 and 1), or based on size (set max_cs > 1).
add_legend	If add_legend = TRUE, add a legend to annotate the size and purity of each CS discovered. It can also be specified as location where legends should be added, e.g., add_legend = "bottomright" (default location is "topright").
...	Additional arguments passed to plot .
L	An integer specifying the number of single-effect rows to plot.
file_prefix	Prefix to path of output plot file. If not specified, the plot, or plots, will be saved to a temporary directory generated using tempdir .

Value

Invisibly returns NULL.

See Also

[susie_plot_changepoint](#)

Examples

```
set.seed(1)
n <- 1000
p <- 1000
beta <- rep(0, p)
beta[sample(1:1000, 4)] <- 1
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
X <- scale(X, center = TRUE, scale = TRUE)
y <- drop(X %*% beta + rnorm(n))
res <- susie(X, y, L = 10)
susie_plot(res, "PIP")
susie_plot(res, "PIP", add_bar = TRUE)
susie_plot(res, "PIP", add_legend = TRUE)
susie_plot(res, "PIP", pos = 1:500, add_legend = TRUE)
# Plot selected regions with adjusted x-axis position label
res$genomic_position <- 1000 + (1:length(res$pip))
susie_plot(res, "PIP",
  add_legend = TRUE,
  pos = list(attr = "genomic_position", start = 1000, end = 1500)
)
# True effects are shown in red.
susie_plot(res, "PIP", b = beta, add_legend = TRUE)
```

```

set.seed(1)
n <- 1000
p <- 1000
beta <- rep(0, p)
beta[sample(1:1000, 4)] <- 1
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
X <- scale(X, center = TRUE, scale = TRUE)
y <- drop(X %*% beta + rnorm(n))
res <- susie(X, y, L = 10, track_fit = TRUE)
susie_plot_iteration(res, L = 10)

```

susie_plot_changepoint

Plot changepoint data and susie fit using ggplot2

Description

Plots original data, y , overlaid with line showing susie fitted value and shaded rectangles showing credible sets for changepoint locations.

Usage

```

susie_plot_changepoint(
  s,
  y,
  line_col = "blue",
  line_size = 1.5,
  cs_col = "red"
)

```

Arguments

<code>s</code>	A susie fit generated by <code>susie_trendfilter(y, order = 0)</code> .
<code>y</code>	An n -vector of observations that are ordered in time or space (assumed equally-spaced).
<code>line_col</code>	Color for the line showing fitted values.
<code>line_size</code>	Size of the lines showing fitted values
<code>cs_col</code>	Color of the shaded rectangles showing credible sets.

Value

A ggplot2 plot object.

Examples

```

set.seed(1)
mu <- c(rep(0, 50), rep(1, 50), rep(3, 50), rep(-2, 50), rep(0, 300))
y <- mu + rnorm(500)
# Here we use a less sensitive tolerance so that the example takes
# less time; in practice you will likely want to use a more stringent
# setting such as tol = 0.001.
s <- susie_trendfilter(y, tol = 0.1)

# Produces ggplot with credible sets for changepoints.
susie_plot_changepoint(s, y)

```

susie_post_outcome_configuration

Post-hoc causal-configuration probabilities for one or more SuSiE-class fits

Description

Runs one of two complementary post-hoc analyses, selected by method: "susiex" (default) for the SuSiEx 2^N combinatorial enumeration, reporting posterior probabilities over binary causality patterns across the N input traits; "coloc_pairwise" for the coloc pairwise ABF, reporting the five colocalisation hypothesis posteriors (H0/H1/H2/H3/H4) for every pair of traits. To get both views, call the function twice and combine.

Usage

```

susie_post_outcome_configuration(
  input,
  by = c("fit", "outcome"),
  outcome_names = NULL,
  method = c("susiex", "coloc_pairwise"),
  prob_thresh = 0.8,
  single_effect_lfsr_cutoff = 0.05,
  cs_only = TRUE,
  p1 = 1e-04,
  p2 = 1e-04,
  p12 = 5e-06,
  ...
)

```

Arguments

input	A single fit of class susie, mvsusie, or mfsusie, OR a list of such fits.
by	Either "fit" (one trait per input fit; default) or "outcome" (multi-output fits expand into per-outcome traits).

outcome_names	Optional character vector of trait-view names. If NULL, names are taken from the input list when available, otherwise <code>trait_1</code> , <code>trait_2</code> , ... are used. If provided, its length must match the number of trait views after applying by.
method	Character scalar; one of "susiex" (default), or "coloc_pairwise". Pick the analysis to run; for both analyses, call the function twice.
prob_thresh	Per-trait marginal threshold for the convenience \$active flags in the SuSiEx output. Default 0.8.
single_effect_lfsr_cutoff	LFSR threshold used to flag outcome-specific CS evidence when method = "susiex" is applied to one multi-output mvsusie or mfsusie fit. The LFSR is read from <code>fit\$lfsr</code> at each CS sentinel SNP. Default 0.05.
cs_only	Logical. If TRUE (default) only enumerate over CSs present in each fit's \$sets\$cs; if FALSE loop over all L rows of \$alpha. Either way, effects whose entire alpha row is zero are skipped. When TRUE, trait views without credible sets are skipped with a warning; if fewer than two trait views remain, the function returns NULL.
p1, p2, p12	Coloc per-SNP causal priors: p1 for trait 1 alone, p2 for trait 2 alone, p12 for shared causal. Defaults match coloc::coloc.bf_bf: p1 = p2 = 1e-4, p12 = 5e-6. Only used when "coloc_pairwise" is in method.
...	Currently ignored.

Details

Two input organizations are supported:

by = "fit" Each input fit contributes a single trait view. Multi-output fits (mvsusie, mfsusie) are kept whole: the trait's per-(CS, SNP) log Bayes factors are the joint composite stored on the fit as `lbf_variable`. Configuration enumeration loops over the cross-product $L_1 \times \dots \times L_N$ of CS indices.

by = "outcome" Multi-output fits fan out into per-outcome views, each with its own per-(CS, SNP) log Bayes factors read from `fit$lbf_variable_outcome` (an $L \times J \times R$ or $L \times J \times M$ array). All per-outcome views share the joint fit's PIP matrix and CS list, so the configuration enumeration reduces to a single index $l \in 1..L$. Single-output susie fits pass through unchanged. Requires `$lbf_variable_outcome` on the fit (set `attach_lbf_variable_outcome = TRUE` when fitting).

After SuSiE has been fit, the post-hoc step can answer two related questions:

"susiex" SuSiEx 2^N activation probabilities for each credible-set tuple. It asks which traits have evidence for a proposed shared CS-level event. In two traits, the both-active state uses same-SNP evidence such as $\sum_j \exp(\ell_{1j} + \ell_{2j})$, where ℓ_{tj} is the SNP-level log Bayes factor for trait t .

"coloc_pairwise" Pairwise coloc H0-H4 posteriors for every trait pair and CS pair. It asks whether same-SNP evidence is stronger than distinct-causal evidence, represented by terms such as $\sum_{j \neq k} \exp(\ell_{1j} + \ell_{2k})$; this separates H3, two distinct causal variants, from H4, one shared causal variant. Per-SNP log Bayes factors are aligned by variant name when available.

Two-trait example: For two traits and one CS pair, SuSiEx has four activation states: (0, 0), (1, 0), (0, 1), and (1, 1). The state (1, 1) means both traits are active for the candidate shared event; it does not introduce a separate distinct-causal state.

Coloc has five hypotheses: H0, no causal variant; H1, trait 1 only; H2, trait 2 only; H3, two distinct causal variants; and H4, one shared causal variant. Thus coloc splits the active two-trait case into H3 and H4, whereas SuSiEx represents it as the single activation state (1, 1).

The "susiex" output is an 2-trait post-hoc activation model over CS tuples. The "coloc_pairwise" output is a pairwise colocalisation model over H0-H4 for each CS pair. Both are summaries of already fitted SuSiE-class models; neither refits the single-trait effects.

More generally, in N -trait analysis, the difference between SuSiEx-style activation and colocalization is a difference in hypothesis space:

SuSiEx activation space "susiex" is an N -trait post-hoc activation, or meta-analysis-style, model over CS tuples. It enumerates the 2^N binary activity patterns and asks which traits participate in a proposed shared event.

Colocalization partition space A theoretical generalization to N -trait colocalization has a larger hypothesis space: inactive traits plus all ways of grouping active traits by shared causal variant, of size $\text{Bell}(N+1)$ (see HyPrColoc, Figure 1). This space asks not only which traits are active, but which active traits share the same causal variant versus distinct causal variants.

Value

NULL if fewer than two trait views are available for post-hoc analysis; otherwise a list of class "susie_post_outcome_configuration" with exactly one of the following components, depending on method:

`$susiex` (when `method = "susiex"`) A list of length equal to the number of CS tuples considered, named `config_1`, `config_2`, etc. Each element has components `config_probability` (binary trait-activation table with a `config_prob` column) and `config_summary` (CS-level post-hoc activation summary, with one row per trait). For one multi-output `mvsusie` or `mfsusie` fit, both `$susiex` and `$mvsusie` are named by CS label, e.g. `L1`; each `$mvsusie` element has `cs_summary`, `config_summary`, and `cs_variant_summary`.

`$coloc_pairwise` (when `method = "coloc_pairwise"`) A `data.frame` with one row per (trait1, trait2, l1, l2) combination, columns `trait1`, `trait2`, `l1`, `l2`, `hit1`, `hit2`, `PP.H0`, `PP.H1`, `PP.H2`, `PP.H3`, `PP.H4`.

Pretty-print with `summary(out)`.

References

SuSiEx, Nature Genetics 2024 (combinatorial 2^N enumeration). Wallace, PLoS Genetics 2020 (coloc pairwise H0/H1/H2/H3/H4 ABF). Foley et al., Nature Communications 2021 (HyPrColoc; multi-trait colocalisation hypothesis space in Figure 1).

Description

Performs SuSiE regression using z-scores and correlation matrix. This is the sufficient-statistics RSS interface. For the specialized regularized eigendecomposition likelihood with $\lambda > 0$, use [susie_rss_lambda](#).

Usage

```
susie_rss(
  z = NULL,
  R = NULL,
  n = NULL,
  X = NULL,
  bhat = NULL,
  shat = NULL,
  var_y = NULL,
  L = min(10, if (is.list(R) && !is.matrix(R)) ncol(R[[1]]) else if (!is.null(R)) ncol(R)
    else if (is.list(X) && !is.matrix(X)) ncol(X[[1]]) else ncol(X)),
  maf = NULL,
  maf_thresh = 0,
  scaled_prior_variance = 0.2,
  residual_variance = NULL,
  prior_weights = NULL,
  null_weight = 0,
  standardize = TRUE,
  estimate_residual_variance = FALSE,
  estimate_residual_method = c("MoM", "MLE", "NIG"),
  z_method = c("wald", "score"),
  estimate_prior_variance = TRUE,
  estimate_prior_method = c("optim", "EM", "simple"),
  prior_variance_grid = NULL,
  mixture_weights = NULL,
  unmappable_effects = c("none", "inf", "ash", "ash_filter_archived"),
  check_null_threshold = 0,
  prior_tol = 1e-09,
  residual_variance_lowerbound = 0,
  residual_variance_upperbound = Inf,
  model_init = NULL,
  s_init = NULL,
  coverage = 0.95,
  min_abs_corr = 0.5,
  max_iter = NULL,
  L_greedy = NULL,
  greedy_lbf_cutoff = 0.1,
```

```

    tol = NULL,
    convergence_method = c("elbo", "pip"),
    verbose = FALSE,
    track_fit = FALSE,
    n_purity = "auto",
    median_abs_corr = NULL,
    cs_extension_corr = NULL,
    refine = FALSE,
    R_finite = NULL,
    R_mismatch = c("none", "eb", "eb_mix"),
    control = susie_rss_control(),
    alpha0 = NULL,
    beta0 = NULL,
    init_only = FALSE,
    slot_prior = NULL
)

```

Arguments

<code>z</code>	A p-vector of z-scores.
<code>R</code>	A p by p correlation matrix. Exactly one of <code>R</code> or <code>X</code> must be provided.
<code>n</code>	The sample size, not required but recommended.
<code>X</code>	A factor matrix ($B \times p$) such that $R = \text{crossprod}(X) / \text{nrow}(X)$ approximates the R (correlation) matrix. When $\text{nrow}(X) \geq \text{ncol}(X)$, the correlation matrix R is formed explicitly and the standard path is used. When $\text{nrow}(X) < \text{ncol}(X)$, a low-rank path is used that avoids forming the $p \times p$ matrix, reducing per-iteration cost from $O(Lp^2)$ to $O(LBp)$. Columns of X are standardized internally.
<code>bhat</code>	Alternative summary data giving the estimated effects (a vector of length p). This, together with <code>shat</code> , may be provided instead of <code>z</code> .
<code>shat</code>	Alternative summary data giving the standard errors of the estimated effects (a vector of length p). This, together with <code>bhat</code> , may be provided instead of <code>z</code> .
<code>var_y</code>	The sample variance of y , defined as $y'y / (n - 1)$. When the sample variance is not provided, the coefficients (returned from <code>coef</code>) are computed on the “standardized” X, y scale.
<code>L</code>	Maximum number of non-zero effects in the model. If <code>L</code> is larger than the number of covariates, p , <code>L</code> is set to p .
<code>maf</code>	A p-vector of minor allele frequencies; to be used along with <code>maf_thresh</code> to filter input summary statistics.
<code>maf_thresh</code>	Variants with MAF smaller than this threshold are not used.
<code>scaled_prior_variance</code>	The prior variance, divided by $\text{var}(y)$ (or by $(1/(n-1))y'y$ for <code>susie_ss</code>); that is, the prior variance of each non-zero element of b is $\text{var}(y) * \text{scaled_prior_variance}$. The value provided should be either a scalar or a vector of length L . If <code>estimate_prior_variance = TRUE</code> , this provides initial estimates of the prior variances.

residual_variance	Variance of the residual. If <code>estimate_residual_variance = TRUE</code> , this value provides the initial estimate of the residual variance. By default, it is set to <code>var(y)</code> in <code>susie</code> and <code>(1/(n-1))yty</code> in <code>susie_ss</code> .
prior_weights	A vector of length <code>p</code> , in which each entry gives the prior probability that corresponding column of <code>X</code> has a nonzero effect on the outcome, <code>y</code> . The weights are internally normalized to sum to 1. When <code>NULL</code> (the default), uniform prior weights are used (each variable is assigned probability $1/p$).
null_weight	Prior probability of no effect (a number between 0 and 1, and cannot be exactly 1).
standardize	If <code>standardize = TRUE</code> , standardize the columns of <code>X</code> to unit variance prior to fitting (or equivalently standardize <code>XtX</code> and <code>Xty</code> to have the same effect). Note that <code>scaled_prior_variance</code> specifies the prior on the coefficients of <code>X</code> <i>after</i> standardization (if it is performed). If you do not standardize, you may need to think more carefully about specifying <code>scaled_prior_variance</code> . Whatever your choice, the coefficients returned by <code>coef</code> are given for <code>X</code> on the original input scale. Any column of <code>X</code> that has zero variance is not standardized.
estimate_residual_variance	The default is <code>FALSE</code> , the residual variance is fixed to 1 or variance of <code>y</code> . If the in-sample <code>R</code> matrix is provided, we recommend setting <code>estimate_residual_variance = TRUE</code> .
estimate_residual_method	The method used for estimating residual variance. For the original SuSiE model, "MLE" and "MoM" estimation is equivalent, but for the infinitesimal model, "MoM" is more stable. We recommend using "NIG" when <code>n < 80</code> for improved coverage, although it is currently only implemented for individual-level data.
z_method	The single-variant test that produced <code>z</code> , either "wald" (default) or "score". With "wald", the z-scores are PVE-adjusted (shrunk by $\sqrt{1 - \text{PVE}_j}$) onto the model's $\sigma^2 = 1$ scale. With "score" (e.g. z-scores from a linear mixed model GWAS), the z-scores are already on that scale, so no adjustment is applied. Forced to "wald" when <code>bhat / shat</code> are supplied, since those are inherently Wald.
estimate_prior_variance	If <code>estimate_prior_variance = TRUE</code> , the prior variance is estimated (this is a separate parameter for each of the <code>L</code> effects). If provided, <code>scaled_prior_variance</code> is then used as an initial value for the optimization. When <code>estimate_prior_variance = FALSE</code> , the prior variance for each of the <code>L</code> effects is determined by the value supplied to <code>scaled_prior_variance</code> .
estimate_prior_method	The method used for estimating prior variance. When <code>estimate_prior_method = "simple"</code> is used, the likelihood at the specified prior variance is compared to the likelihood at a variance of zero, and the setting with the larger likelihood is retained. When <code>prior_variance_grid</code> is provided, this is automatically set to "fixed_mixture".
prior_variance_grid	Numeric vector of <code>K</code> prior variances defining a mixture-of-normals prior on effect sizes. When provided, the SER evaluates Bayes factors at each grid point

	and forms a mixture BF weighted by <code>mixture_weights</code> . This bypasses the scalar prior variance optimization. Default is NULL (standard scalar V path).
<code>mixture_weights</code>	Numeric vector of K non-negative weights summing to 1, giving the mixture proportions for the variance grid. Default is NULL, which uses uniform weights when <code>prior_variance_grid</code> is provided.
<code>unmappable_effects</code>	The method for modeling unmappable effects: "none", "inf", "ash".
<code>check_null_threshold</code>	When the prior variance is estimated, compare the estimate with the null, and set the prior variance to zero unless the log-likelihood using the estimate is larger by this threshold amount. For example, if you set <code>check_null_threshold = 0.1</code> , this will "nudge" the estimate towards zero when the difference in log-likelihoods is small. A note of caution that setting this to a value greater than zero may lead the IBSS fitting procedure to occasionally decrease the ELBO. This setting is disabled when using <code>unmappable_effects = "inf"</code> or <code>unmappable_effects = "ash"</code> .
<code>prior_tol</code>	When the prior variance is estimated, compare the estimated value to <code>prior_tol</code> at the end of the computation, and exclude a single effect from PIP computation if the estimated prior variance is smaller than this tolerance value.
<code>residual_variance_lowerbound</code>	Lower limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
<code>residual_variance_upperbound</code>	Upper limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
<code>model_init</code>	A previous susie fit with which to initialize.
<code>s_init</code>	Deprecated alias for <code>model_init</code> .
<code>coverage</code>	A number between 0 and 1 specifying the "coverage" of the estimated confidence sets.
<code>min_abs_corr</code>	Minimum absolute correlation allowed in a credible set. The default, 0.5, corresponds to a squared correlation of 0.25, which is a commonly used threshold for genotype data in genetic studies. This "purity" filter is applied to the CSs reported in the fit object, so the CS list returned here may be a subset of the one produced by calling <code>susie_get_cs</code> on the same fit without passing X or Xcorr (in which case the purity filter is skipped). Set to NULL to disable the minimum-correlation clause (e.g. to filter on <code>median_abs_corr</code> alone).
<code>max_iter</code>	Maximum number of IBSS iterations to perform. For <code>susie_rss()</code> and <code>susie_rss_lambda()</code> , NULL uses 50 with a hint; other interfaces use 100.
<code>L_greedy</code>	Integer or NULL. When non-NULL, run a greedy outer loop that grows the number of effects from <code>L_greedy</code> up to L in linear steps until the fit saturates. The default NULL runs the usual fixed-L fit.
<code>greedy_lbf_cutoff</code>	Numeric saturation threshold for the <code>L_greedy</code> outer loop. Default is 0.1.

tol	A small, non-negative number specifying the convergence tolerance for the IBSS fitting procedure. When NULL, the default is 1e-4; for <code>estimate_residual_method = "NIG"</code> , the default is tightened to 1e-6.
convergence_method	When <code>converge_method = "elbo"</code> the fitting procedure halts when the difference in the variational lower bound, or “ELBO” (the objective function to be maximized), is less than <code>tol</code> . When <code>converge_method = "pip"</code> the fitting procedure halts when the maximum absolute difference in alpha is less than <code>tol</code> .
verbose	If <code>verbose = TRUE</code> , the algorithm’s progress, a summary of the optimization settings, and refinement progress (if <code>refine = TRUE</code>) are printed to the console.
track_fit	If <code>track_fit = TRUE</code> , a compact <code>susie_track</code> object is returned in <code>trace</code> , containing alpha history, effect summaries and available diagnostics at each iteration of the IBSS fitting procedure.
n_purity	Passed as argument <code>n_purity</code> to susie_get_cs .
median_abs_corr	An optional second purity threshold applied to the median absolute correlation among the CS variables (default NULL, off). When both <code>min_abs_corr</code> and <code>median_abs_corr</code> are set, they are OR-linked: a CS is kept if it clears either one. See susie_get_cs .
cs_extension_corr	Passed as argument <code>cs_extension_corr</code> to susie_get_cs when the credible sets are constructed. Either NULL (the default, meaning no correlation-based extension) or a single number between 0 and 1. When set, each credible set is extended to absorb every variable whose absolute correlation with a set member exceeds the threshold (near-perfectly correlated proxies); 0.99 is the recommended value.
refine	If <code>refine = TRUE</code> , then an additional iterative refinement procedure is used, after the IBSS algorithm, to check and escape from local optima (see details).
R_finite	Controls variance inflation to account for estimating the R matrix from a finite reference panel. Accepts four types of input: NULL (default) The R matrix is treated as trusted, and no finite-reference variance inflation is applied. With <code>estimate_residual_variance = TRUE</code> , this keeps the in-sample R warning active. FALSE No finite-reference variance inflation is applied, and the in-sample R warning for <code>estimate_residual_variance = TRUE</code> is silenced. Use this only when R is the in-sample correlation matrix. TRUE Infer the reference sample size B from the input X. Sets <code>B = nrow(X)</code> for single-panel input, or one B per panel for multi-panel input. Requires X to be provided (errors if only R is given, since B cannot be inferred). Number Explicit reference sample size B. When active, this dynamically inflates the null variance of each variable’s score statistic at every IBSS iteration to account for finite-reference uncertainty in the Single Effect Regression (SER). When provided, the output includes a <code>R_finite_diagnostics</code> element with per-region and per-variable quality metrics.

R_mismatch	R-mismatch correction mode. "none" (default) is off. "eb" is the recommended empirical Bayes procedure for mismatch correction described in Sun et al. (2026+). It updates a region-level variance component during the IBSS iterations and reports a QC score (Q_art) that extends the Zou et al. (2022) column-space check from the input summary vector to the fitted residual after correction. It warns when that residual still projects onto near-null directions of the supplied R, and auto-disables estimate_residual_variance with a warning. "eb_mix" estimates the same regional component as "eb", but applies the population term locally as $w_j * \lambda$, where w_j is a per-variant mismatch probability from a two-component residual mixture. The finite-reference term B^{-1} is unchanged.
control	Advanced RSS numerical and diagnostic settings created by susie_rss_control . A partial named list is also accepted.
alpha0	Numerical parameter for the NIG prior when using estimate_residual_method = "NIG". Defaults to $1/\sqrt{n}$, where n is the sample size. When calling susie_rss with NIG, n must be supplied; otherwise validation errors.
beta0	Numerical parameter for the NIG prior when using estimate_residual_method = "NIG". Defaults to $1/\sqrt{n}$, where n is the sample size. When calling susie_rss with NIG, n must be supplied; otherwise validation errors.
init_only	Logical. If TRUE, return a list with data and params objects without running the IBSS algorithm. Default is FALSE.
slot_prior	Optional slot activity prior created by slot_prior_betabinom or slot_prior_poisson . Use slot_prior_betabinom(a_beta, b_beta) for the usual single-locus setting; it places a Beta-Binomial prior on the number of active effects and gives an adaptive multiplicity correction. Use slot_prior_poisson(C, nu) when you want a Gamma-Poisson prior centered on an expected number C of active effects. When supplied, each single-effect slot has an estimated activity probability c_{hat} ; fitted values and PIPs are weighted by these activity probabilities, and convergence is checked using convergence_method = "pip".

Details

Multi-panel input: R may be a list of K correlation matrices, or X a list of K factor matrices ($B_k \times p$), one per reference panel. The fit has two stages. First, the panel weights ω are estimated from the data. Then `susie_rss` is fit on the fixed weighted reference $R(\omega) = \sum_k \omega_k R_k$. For X input this is the stacked factor matrix $X(\omega) = [\sqrt{\omega_1} \tilde{X}_1; \dots; \sqrt{\omega_K} \tilde{X}_K]$, where $\tilde{X}_k$ is X_k with columns standardized so that $\tilde{X}_k^\top \tilde{X}_k = R_k$; then $X(\omega)^\top X(\omega) = R(\omega)$. When `R_finite` is set, the refit uses $B_{\text{eff}} = 1 / \sum_k \omega_k^2 / B_k$.

Value

In addition to the standard "susie" output (see [susie](#)), the returned object may contain:

R_finite_diagnostics

A list of diagnostics for the R-uncertainty correction (only present when `R_finite` is provided or `R_mismatch != "none"`), containing: B (the reference sample size); p (number of variables); effective_rank (debaised $\tilde{r} = p^2 / \|R\|_F^2$); r_over_B ($\tilde{r}/B$, one number per region; values ≤ 0.2 indicate the reference

	panel is adequate); Rhat_diag_deviation ($ \hat{R}_{jj} - 1 $, one number per variable); lambda_bias (region-level scalar on the default lambda = 0 sufficient-statistics path when R_mismatch != "none"); B_corrected (effective reference sample size after the R-mismatch correction, $1/(1/B + \lambda_{\text{bias}})$; substantially smaller than the input B flags a dominant population mismatch component); per_variable_penalty (final-iteration $v_j/\sigma^2 = \tau_j^2/\sigma^2 - 1$, one number per variable; values ≤ 0.2 indicate minimal power loss, values $\gg 1$ flag variables where the correction is doing heavy lifting); R_reliability_flag (TRUE when artifact_flag (Q_art > artifact_threshold) or R_sensitivity_flag (a credible set's log-BF attenuation $\geq$ sensitivity_threshold) is TRUE); SER_fallback (identical to R_reliability_flag; TRUE means the one-effect model in ser_model should be used instead of the full fit).
omega_weights	For multi-panel input, the estimated panel weights ω , in the order the panels were supplied. The rest of the fit is a single-reference fit against $\sum_k \omega_k R_k$.
single_panel_fits	For multi-panel input, the K single-panel fits used to choose the panel that initializes the mixture.

susie_rss_control	<i>Control parameters for SuSiE RSS</i>
-------------------	---

Description

Control parameters for SuSiE RSS

Usage

```
susie_rss_control(
  mismatch_estimator = c("mle", "map"),
  mixture_reference_p = 5e-08,
  qc_eigen_tol_rel = 0.001,
  qc_eigen_tol_abs = 0,
  artifact_threshold = 0.1,
  sensitivity_threshold = 30,
  r_tol = 1e-08,
  check_input = FALSE,
  check_prior = TRUE
)
```

Arguments

mismatch_estimator	Estimator for the region-level mismatch variance: "mle" or "map".
mixture_reference_p	Two-sided reference P-value used to calibrate prior odds for R_mismatch = "eb_mix".

qc_eigen_tol_rel, qc_eigen_tol_abs	Relative and absolute cutoffs for low-eigenvalue directions in the RSS mismatch diagnostic.
artifact_threshold	Threshold in $[0, 1]$ for the residual artifact score.
sensitivity_threshold	Threshold for credible-set Bayes-factor attenuation.
r_tol	Tolerance for positive-semidefinite matrix checks.
check_input	Whether to check summary-statistic matrices for positive semidefiniteness and consistency.
check_prior	Whether to check for an unreasonably large estimated prior variance relative to the marginal z-scores.

Value

A named list for the control argument of [susie_rss](#).

susie_rss_lambda	<i>Sum of Single Effects Regression using the RSS-lambda likelihood</i>
------------------	---

Description

Specialized interface for the regularized eigendecomposition RSS likelihood of Zou et al. (2022). This path accepts a single reference matrix or a single factor matrix and does not support multi-panel mixture, finite-reference inflation, or R-mismatch correction.

Usage

```
susie_rss_lambda(
  z = NULL,
  R = NULL,
  n = NULL,
  X = NULL,
  L = min(10, if (!is.null(R)) ncol(R) else ncol(X)),
  lambda,
  maf = NULL,
  maf_thresh = 0,
  prior_variance = 50,
  residual_variance = NULL,
  prior_weights = NULL,
  null_weight = 0,
  intercept_value = 0,
  estimate_residual_variance = FALSE,
  estimate_residual_method = "MLE",
  z_method = c("wald", "score"),
  estimate_prior_variance = TRUE,
```

```

estimate_prior_method = c("optim", "EM", "simple"),
prior_variance_grid = NULL,
mixture_weights = NULL,
check_null_threshold = 0,
prior_tol = 1e-09,
residual_variance_lowerbound = 0,
model_init = NULL,
coverage = 0.95,
min_abs_corr = 0.5,
max_iter = NULL,
L_greedy = NULL,
greedy_lbf_cutoff = 0.1,
tol = NULL,
convergence_method = c("elbo", "pip"),
verbose = FALSE,
track_fit = FALSE,
check_prior = TRUE,
check_R = TRUE,
check_z = FALSE,
n_purity = "auto",
median_abs_corr = NULL,
cs_extension_corr = NULL,
r_tol = 1e-08,
refine = FALSE,
init_only = FALSE,
slot_prior = NULL
)

```

Arguments

z	A p-vector of z-scores.
R	A p by p correlation matrix. Exactly one of R or X must be provided.
n	The sample size, not required but recommended.
X	Optional factor matrix (B by p) such that $R = \text{crossprod}(X) / \text{nrow}(X)$ approximates the correlation matrix R. The RSS-lambda path always builds the eigen decomposition via SVD of standardized X; there is no separate low-rank branch. Provide either R or X, not both.
L	Maximum number of non-zero effects in the model. If L is larger than the number of covariates, p, L is set to p.
lambda	Regularization parameter for the RSS-lambda likelihood. Must be supplied. lambda = "estimate" estimates lambda from the null-space residual.
maf	A p-vector of minor allele frequencies; to be used along with maf_thresh to filter input summary statistics.
maf_thresh	Variants with MAF smaller than this threshold are not used.
prior_variance	Prior variance for each non-zero effect on the z-score scale. Replaces scaled_prior_variance from susie_rss . Default 50.

<code>residual_variance</code>	Residual variance on the RSS-lambda scale. If NULL, it is initialized to $1 - \lambda$; if supplied, the working residual variance is <code>residual_variance - lambda</code> .
<code>prior_weights</code>	A vector of length p , in which each entry gives the prior probability that corresponding column of X has a nonzero effect on the outcome, y . The weights are internally normalized to sum to 1. When NULL (the default), uniform prior weights are used (each variable is assigned probability $1/p$).
<code>null_weight</code>	Prior probability of no effect (a number between 0 and 1, and cannot be exactly 1).
<code>intercept_value</code>	Intercept used by the RSS-lambda likelihood. Default 0.
<code>estimate_residual_variance</code>	If TRUE, estimate the RSS-lambda residual variance by maximum likelihood.
<code>estimate_residual_method</code>	Variance-component estimator. The RSS-lambda path supports "MLE" only; any other value errors.
<code>z_method</code>	The single-variant test that produced z , either "wald" (default; z -scores are PVE-adjusted onto the $\sigma^2 = 1$ scale) or "score" (z -scores already on that scale, e.g. from a linear mixed model GWAS, so no adjustment is applied).
<code>estimate_prior_variance</code>	If <code>estimate_prior_variance = TRUE</code> , the prior variance is estimated (a separate parameter for each of the L effects). When TRUE, <code>prior_variance</code> provides the initial value; when FALSE, it is held fixed.
<code>estimate_prior_method</code>	The method used for estimating prior variance. When <code>estimate_prior_method = "simple"</code> is used, the likelihood at the specified prior variance is compared to the likelihood at a variance of zero, and the setting with the larger likelihood is retained. When <code>prior_variance_grid</code> is provided, this is automatically set to "fixed_mixture".
<code>prior_variance_grid</code>	Numeric vector of K prior variances defining a mixture-of-normals prior on effect sizes. When provided, the SER evaluates Bayes factors at each grid point and forms a mixture BF weighted by <code>mixture_weights</code> . This bypasses the scalar prior variance optimization. Default is NULL (standard scalar V path).
<code>mixture_weights</code>	Numeric vector of K non-negative weights summing to 1, giving the mixture proportions for the variance grid. Default is NULL, which uses uniform weights when <code>prior_variance_grid</code> is provided.
<code>check_null_threshold</code>	When the prior variance is estimated, compare its likelihood to the likelihood at zero and use zero unless the larger value exceeds it by at least <code>check_null_threshold</code> . 0 (default) takes the larger likelihood at face value.
<code>prior_tol</code>	When the prior variance is estimated, compare the estimated value to <code>prior_tol</code> at the end of the computation, and exclude a single effect from PIP computation if the estimated prior variance is smaller than this tolerance value.

residual_variance_lowerbound	Lower limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
model_init	A previous susie fit with which to initialize.
coverage	A number between 0 and 1 specifying the “coverage” of the estimated confidence sets.
min_abs_corr	Minimum absolute correlation allowed in a credible set. The default, 0.5, corresponds to a squared correlation of 0.25, which is a commonly used threshold for genotype data in genetic studies. This “purity” filter is applied to the CSs reported in the fit object, so the CS list returned here may be a subset of the one produced by calling susie_get_cs on the same fit without passing <code>X</code> or <code>Xcorr</code> (in which case the purity filter is skipped). Set to <code>NULL</code> to disable the minimum-correlation clause (e.g. to filter on <code>median_abs_corr</code> alone).
max_iter	Maximum number of IBSS iterations to perform. For <code>susie_rss()</code> and <code>susie_rss_lambda()</code> , <code>NULL</code> uses 50 with a hint; other interfaces use 100.
L_greedy	Integer or <code>NULL</code> . When non- <code>NULL</code> , run a greedy outer loop that grows the number of effects from <code>L_greedy</code> up to <code>L</code> in linear steps until the fit saturates. The default <code>NULL</code> runs the usual fixed- <code>L</code> fit.
greedy_lbf_cutoff	Numeric saturation threshold for the <code>L_greedy</code> outer loop. Default is 0.1.
tol	A small, non-negative number specifying the convergence tolerance for the IBSS fitting procedure. When <code>NULL</code> , the default is $1e-4$; for <code>estimate_residual_method = "NIG"</code> , the default is tightened to $1e-6$.
convergence_method	When <code>converge_method = "elbo"</code> the fitting procedure halts when the difference in the variational lower bound, or “ELBO” (the objective function to be maximized), is less than <code>tol</code> . When <code>converge_method = "pip"</code> the fitting procedure halts when the maximum absolute difference in <code>alpha</code> is less than <code>tol</code> .
verbose	If <code>verbose = TRUE</code> , the algorithm’s progress, a summary of the optimization settings, and refinement progress (if <code>refine = TRUE</code>) are printed to the console.
track_fit	If <code>track_fit = TRUE</code> , a compact <code>susie_track</code> object is returned in <code>trace</code> , containing <code>alpha</code> history, effect summaries and available diagnostics at each iteration of the IBSS fitting procedure.
check_prior	If <code>check_prior = TRUE</code> , it checks if the estimated prior variance becomes unreasonably large (comparing with $10 * \max(\text{abs}(z))^2$).
check_R	If <code>TRUE</code> , verify that <code>R</code> is positive semidefinite.
check_z	If <code>TRUE</code> , verify that <code>z</code> lies in the column space of <code>R</code> .
n_purity	Passed as argument <code>n_purity</code> to susie_get_cs .
median_abs_corr	An optional second purity threshold applied to the median absolute correlation among the CS variables (default <code>NULL</code> , off). When both <code>min_abs_corr</code> and <code>median_abs_corr</code> are set, they are OR-linked: a CS is kept if it clears either one. See susie_get_cs .

<code>cs_extension_corr</code>	Passed as argument <code>cs_extension_corr</code> to susie_get_cs when the credible sets are constructed. Either <code>NULL</code> (the default, meaning no correlation-based extension) or a single number between 0 and 1. When set, each credible set is extended to absorb every variable whose absolute correlation with a set member exceeds the threshold (near-perfectly correlated proxies); 0.99 is the recommended value.
<code>r_tol</code>	Tolerance level for eigenvalue check of positive semidefinite matrix XtX .
<code>refine</code>	If <code>refine = TRUE</code> , then an additional iterative refinement procedure is used, after the IBSS algorithm, to check and escape from local optima (see details).
<code>init_only</code>	Logical. If <code>TRUE</code> , return a list with data and params objects without running the IBSS algorithm. Default is <code>FALSE</code> .
<code>slot_prior</code>	Optional slot activity prior created by slot_prior_betabinom or slot_prior_poisson . Use <code>slot_prior_betabinom(a_beta, b_beta)</code> for the usual single-locus setting; it places a Beta-Binomial prior on the number of active effects and gives an adaptive multiplicity correction. Use <code>slot_prior_poisson(C, nu)</code> when you want a Gamma-Poisson prior centered on an expected number <code>C</code> of active effects. When supplied, each single-effect slot has an estimated activity probability <code>c_hat</code> ; fitted values and PIPs are weighted by these activity probabilities, and convergence is checked using <code>convergence_method = "pip"</code> .

Value

A "susie" fit (or, with `init_only = TRUE`, the constructed data and params objects).

<code>susie_ser</code>	<i>Single-effect regression from summary statistics</i>
------------------------	---

Description

Fit a single-effect regression (SER) model directly from marginal summary statistics. Unlike [susie_rss](#), this interface does not take an LD matrix or reference factor matrix, and it never constructs a diagonal LD matrix internally. The model has exactly one single effect.

Usage

```
susie_ser(
  z = NULL,
  bhat = NULL,
  shat = NULL,
  n = NULL,
  var_y = NULL,
  prior_variance = 50,
  scaled_prior_variance = 0.2,
  prior_weights = NULL,
  null_weight = 0,
```

```

estimate_prior_method = c("optim", "simple"),
check_null_threshold = 0,
prior_tol = 1e-09,
coverage = 0.95,
ethres = NULL
)

```

Arguments

<code>z</code>	A vector of z-scores. Provide either <code>z</code> , or both <code>bhat</code> and <code>shat</code> , but not both.
<code>bhat</code>	Alternative summary statistics: marginal effect estimates.
<code>shat</code>	Standard errors for <code>bhat</code> ; may be a scalar or a vector with the same length as <code>bhat</code> .
<code>n</code>	Optional sample size. When supplied, z-scores are adjusted using the same PVE adjustment as susie_rss .
<code>var_y</code>	Optional sample variance of the outcome. When supplied with <code>n</code> , this sets the working outcome scale. With <code>bhat</code> and <code>shat</code> , coefficients returned by coef.susie are on the original <code>bhat</code> scale.
<code>prior_variance</code>	Prior variance on the z-score scale when <code>n</code> is not supplied. Default 50, matching the z-only RSS scale.
<code>scaled_prior_variance</code>	Prior variance divided by the working outcome variance when <code>n</code> is supplied. Default 0.2.
<code>prior_weights</code>	A vector of prior probabilities, one per variable. The weights are normalized internally.
<code>null_weight</code>	Prior probability of no effect. If nonzero, a null variable is appended with zero effect.
<code>estimate_prior_method</code>	Method used for the single prior variance. "optim" optimizes the one-effect marginal likelihood; "simple" uses the supplied prior variance and applies the null-threshold check.
<code>check_null_threshold</code>	If the log-likelihood at prior variance zero is within this amount of the selected prior variance, set the prior variance to zero.
<code>prior_tol</code>	Prior-variance tolerance used by susie_get_pip .
<code>coverage</code>	Coverage for credible-set construction. If NULL, credible sets are not constructed and no no-LD hint is emitted.
<code>ethres</code>	Effective-number threshold passed to susie_get_cs_attainable .

Value

A "susie" object with one single effect.

See Also

[susie_rss](#), [susie_get_cs_attainable](#), [susie_get_cs](#)

susie_ss

SuSiE using Sufficient Statistics

Description

Performs SuSiE regression using sufficient statistics (XtX , Xty , yty , n) instead of individual-level data (X , y).

Usage

```
susie_ss(
  XtX,
  Xty,
  yty,
  n,
  L = min(10, ncol(XtX)),
  X_colmeans = NA,
  y_mean = NA,
  maf = NULL,
  maf_thresh = 0,
  check_input = FALSE,
  r_tol = 1e-08,
  standardize = TRUE,
  scaled_prior_variance = 0.2,
  residual_variance = NULL,
  prior_weights = NULL,
  null_weight = 0,
  model_init = NULL,
  s_init = NULL,
  estimate_residual_variance = TRUE,
  estimate_residual_method = c("MoM", "MLE", "NIG"),
  residual_variance_lowerbound = 0,
  residual_variance_upperbound = Inf,
  estimate_prior_variance = TRUE,
  estimate_prior_method = c("optim", "EM", "simple"),
  prior_variance_grid = NULL,
  mixture_weights = NULL,
  unmappable_effects = c("none", "inf", "ash", "ash_filter_archived"),
  check_null_threshold = 0,
  prior_tol = 1e-09,
  max_iter = 100,
  L_greedy = NULL,
  greedy_lbf_cutoff = 0.1,
  tol = NULL,
  convergence_method = c("elbo", "pip"),
  coverage = 0.95,
  min_abs_corr = 0.5,
```

```

    n_purity = "auto",
    median_abs_corr = NULL,
    cs_extension_corr = NULL,
    verbose = FALSE,
    track_fit = FALSE,
    check_prior = FALSE,
    refine = FALSE,
    alpha0 = NULL,
    beta0 = NULL,
    slot_prior = NULL
)

```

Arguments

<code>XtX</code>	A p by p matrix, $X'X$, with columns of X centered to have mean zero.
<code>Xty</code>	A p -vector, $X'y$, with y and columns of X centered to have mean zero.
<code>yty</code>	A scalar, $y'y$, with y centered to have mean zero.
<code>n</code>	The sample size.
<code>L</code>	Maximum number of non-zero effects in the model. If L is larger than the number of covariates, p , L is set to p .
<code>X_colmeans</code>	A p -vector of column means of X . If both <code>X_colmeans</code> and <code>y_mean</code> are provided, the intercept is estimated; otherwise, the intercept is NA.
<code>y_mean</code>	A scalar containing the mean of y . If both <code>X_colmeans</code> and <code>y_mean</code> are provided, the intercept is estimated; otherwise, the intercept is NA.
<code>maf</code>	A p -vector of minor allele frequencies; to be used along with <code>maf_thresh</code> to filter input summary statistics.
<code>maf_thresh</code>	Variants with MAF smaller than this threshold are not used.
<code>check_input</code>	If <code>check_input = TRUE</code> , <code>susie_ss</code> performs additional checks on <code>XtX</code> and <code>Xty</code> . The checks are: (1) check that <code>XtX</code> is positive semidefinite; (2) check that <code>Xty</code> is in the space spanned by the non-zero eigenvectors of <code>XtX</code> .
<code>r_tol</code>	Tolerance level for eigenvalue check of positive semidefinite matrix <code>XtX</code> .
<code>standardize</code>	If <code>standardize = TRUE</code> , standardize the columns of X to unit variance prior to fitting (or equivalently standardize <code>XtX</code> and <code>Xty</code> to have the same effect). Note that <code>scaled_prior_variance</code> specifies the prior on the coefficients of X <i>after</i> standardization (if it is performed). If you do not standardize, you may need to think more carefully about specifying <code>scaled_prior_variance</code> . Whatever your choice, the coefficients returned by <code>coef</code> are given for X on the original input scale. Any column of X that has zero variance is not standardized.
<code>scaled_prior_variance</code>	The prior variance, divided by $\text{var}(y)$ (or by $(1/(n-1))yty$ for <code>susie_ss</code>); that is, the prior variance of each non-zero element of b is $\text{var}(y) * \text{scaled_prior_variance}$. The value provided should be either a scalar or a vector of length L . If <code>estimate_prior_variance = TRUE</code> , this provides initial estimates of the prior variances.

<code>residual_variance</code>	Variance of the residual. If <code>estimate_residual_variance = TRUE</code> , this value provides the initial estimate of the residual variance. By default, it is set to $\text{var}(y)$ in <code>susie</code> and $(1/(n-1))y^T y$ in <code>susie_ss</code> .
<code>prior_weights</code>	A vector of length p , in which each entry gives the prior probability that corresponding column of X has a nonzero effect on the outcome, y . The weights are internally normalized to sum to 1. When <code>NULL</code> (the default), uniform prior weights are used (each variable is assigned probability $1/p$).
<code>null_weight</code>	Prior probability of no effect (a number between 0 and 1, and cannot be exactly 1).
<code>model_init</code>	A previous <code>susie</code> fit with which to initialize.
<code>s_init</code>	Deprecated alias for <code>model_init</code> .
<code>estimate_residual_variance</code>	If <code>estimate_residual_variance = TRUE</code> , the residual variance is estimated, using <code>residual_variance</code> as an initial value. If <code>estimate_residual_variance = FALSE</code> , the residual variance is fixed to the value supplied by <code>residual_variance</code> .
<code>estimate_residual_method</code>	The method used for estimating residual variance. For the original SuSiE model, "MLE" and "MoM" estimation is equivalent, but for the infinitesimal model, "MoM" is more stable. We recommend using "NIG" when $n < 80$ for improved coverage, although it is currently only implemented for individual-level data.
<code>residual_variance_lowerbound</code>	Lower limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
<code>residual_variance_upperbound</code>	Upper limit on the estimated residual variance. It is only relevant when <code>estimate_residual_variance = TRUE</code> .
<code>estimate_prior_variance</code>	If <code>estimate_prior_variance = TRUE</code> , the prior variance is estimated (this is a separate parameter for each of the L effects). If provided, <code>scaled_prior_variance</code> is then used as an initial value for the optimization. When <code>estimate_prior_variance = FALSE</code> , the prior variance for each of the L effects is determined by the value supplied to <code>scaled_prior_variance</code> .
<code>estimate_prior_method</code>	The method used for estimating prior variance. When <code>estimate_prior_method = "simple"</code> is used, the likelihood at the specified prior variance is compared to the likelihood at a variance of zero, and the setting with the larger likelihood is retained. When <code>prior_variance_grid</code> is provided, this is automatically set to "fixed_mixture".
<code>prior_variance_grid</code>	Numeric vector of K prior variances defining a mixture-of-normals prior on effect sizes. When provided, the SER evaluates Bayes factors at each grid point and forms a mixture BF weighted by <code>mixture_weights</code> . This bypasses the scalar prior variance optimization. Default is <code>NULL</code> (standard scalar V path).
<code>mixture_weights</code>	Numeric vector of K non-negative weights summing to 1, giving the mixture proportions for the variance grid. Default is <code>NULL</code> , which uses uniform weights when <code>prior_variance_grid</code> is provided.

unmappable_effects	The method for modeling unmappable effects: "none", "inf", "ash".
check_null_threshold	When the prior variance is estimated, compare the estimate with the null, and set the prior variance to zero unless the log-likelihood using the estimate is larger by this threshold amount. For example, if you set <code>check_null_threshold = 0.1</code> , this will "nudge" the estimate towards zero when the difference in log-likelihoods is small. A note of caution that setting this to a value greater than zero may lead the IBSS fitting procedure to occasionally decrease the ELBO. This setting is disabled when using <code>unmappable_effects = "inf"</code> or <code>unmappable_effects = "ash"</code> .
prior_tol	When the prior variance is estimated, compare the estimated value to <code>prior_tol</code> at the end of the computation, and exclude a single effect from PIP computation if the estimated prior variance is smaller than this tolerance value.
max_iter	Maximum number of IBSS iterations to perform. For <code>susie_rss()</code> and <code>susie_rss_lambda()</code> , NULL uses 50 with a hint; other interfaces use 100.
L_greedy	Integer or NULL. When non-NULL, run a greedy outer loop that grows the number of effects from <code>L_greedy</code> up to <code>L</code> in linear steps until the fit saturates. The default NULL runs the usual fixed-L fit.
greedy_lbf_cutoff	Numeric saturation threshold for the <code>L_greedy</code> outer loop. Default is 0.1.
tol	A small, non-negative number specifying the convergence tolerance for the IBSS fitting procedure. When NULL, the default is $1e-4$; for <code>estimate_residual_method = "NIG"</code> , the default is tightened to $1e-6$.
convergence_method	When <code>converge_method = "elbo"</code> the fitting procedure halts when the difference in the variational lower bound, or "ELBO" (the objective function to be maximized), is less than <code>tol</code> . When <code>converge_method = "pip"</code> the fitting procedure halts when the maximum absolute difference in alpha is less than <code>tol</code> .
coverage	A number between 0 and 1 specifying the "coverage" of the estimated confidence sets.
min_abs_corr	Minimum absolute correlation allowed in a credible set. The default, 0.5, corresponds to a squared correlation of 0.25, which is a commonly used threshold for genotype data in genetic studies. This "purity" filter is applied to the CSs reported in the fit object, so the CS list returned here may be a subset of the one produced by calling <code>susie_get_cs</code> on the same fit without passing <code>X</code> or <code>Xcorr</code> (in which case the purity filter is skipped). Set to NULL to disable the minimum-correlation clause (e.g. to filter on <code>median_abs_corr</code> alone).
n_purity	Passed as argument <code>n_purity</code> to <code>susie_get_cs</code> .
median_abs_corr	An optional second purity threshold applied to the median absolute correlation among the CS variables (default NULL, off). When both <code>min_abs_corr</code> and <code>median_abs_corr</code> are set, they are OR-linked: a CS is kept if it clears either one. See <code>susie_get_cs</code> .

<code>cs_extension_corr</code>	Passed as argument <code>cs_extension_corr</code> to susie_get_cs when the credible sets are constructed. Either NULL (the default, meaning no correlation-based extension) or a single number between 0 and 1. When set, each credible set is extended to absorb every variable whose absolute correlation with a set member exceeds the threshold (near-perfectly correlated proxies); 0.99 is the recommended value.
<code>verbose</code>	If <code>verbose = TRUE</code> , the algorithm's progress, a summary of the optimization settings, and refinement progress (if <code>refine = TRUE</code>) are printed to the console.
<code>track_fit</code>	If <code>track_fit = TRUE</code> , a compact <code>susie_track</code> object is returned in <code>trace</code> , containing alpha history, effect summaries and available diagnostics at each iteration of the IBSS fitting procedure.
<code>check_prior</code>	If <code>check_prior = TRUE</code> , it checks if the estimated prior variance becomes unreasonably large (comparing with $10 * \max(\text{abs}(z))^2$).
<code>refine</code>	If <code>refine = TRUE</code> , then an additional iterative refinement procedure is used, after the IBSS algorithm, to check and escape from local optima (see details).
<code>alpha0</code>	Numerical parameter for the NIG prior when using <code>estimate_residual_method = "NIG"</code> . Defaults to $1/\sqrt{n}$, where n is the sample size. When calling <code>susie_rss</code> with NIG, n must be supplied; otherwise validation errors.
<code>beta0</code>	Numerical parameter for the NIG prior when using <code>estimate_residual_method = "NIG"</code> . Defaults to $1/\sqrt{n}$, where n is the sample size. When calling <code>susie_rss</code> with NIG, n must be supplied; otherwise validation errors.
<code>slot_prior</code>	Optional slot activity prior created by slot_prior_betabinom or slot_prior_poisson . Use <code>slot_prior_betabinom(a_beta, b_beta)</code> for the usual single-locus setting; it places a Beta-Binomial prior on the number of active effects and gives an adaptive multiplicity correction. Use <code>slot_prior_poisson(C, nu)</code> when you want a Gamma-Poisson prior centered on an expected number C of active effects. When supplied, each single-effect slot has an estimated activity probability $\hat{c}$; fitted values and PIPs are weighted by these activity probabilities, and convergence is checked using <code>convergence_method = "pip"</code> .

<code>susie_trendfilter</code>	<i>Apply susie to trend filtering (especially changepoint problems), a type of non-parametric regression.</i>
--------------------------------	---

Description

Fits the non-parametric Gaussian regression model $y = \mu + e$, where the mean μ is modelled as $\mu = Xb$, X is a matrix with columns containing an appropriate basis, and b is vector with a (sparse) SuSiE prior. In particular, when `order = 0`, the j th column of X is a vector with the first j elements equal to zero, and the remaining elements equal to 1, so that b_j corresponds to the change in the mean of y between indices j and $j+1$. For background on trend filtering, see Tibshirani (2014). See also the "Trend filtering" vignette, `vignette("trend_filtering")`.

Usage

```
susie_trendfilter(y, order = 0, standardize = FALSE, use_mad = TRUE, ...)
```

Arguments

<code>y</code>	An n-vector of observations ordered in time or space (assumed to be equally spaced).
<code>order</code>	An integer specifying the order of trend filtering. The default, <code>order = 0</code> , corresponds to "change point" problems (<i>i.e.</i> , piecewise constant μ). Although <code>order > 0</code> is implemented, we do not recommend its use; in practice, we have found problems with convergence of the algorithm to poor local optima, producing unreliable inferences.
<code>standardize</code>	Logical indicating whether to standardize the X variables ("basis functions"); <code>standardize = FALSE</code> is recommended as these basis functions already have a natural scale.
<code>use_mad</code>	Logical indicating whether to use the "median absolute deviation" (MAD) method to the estimate residual variance. If <code>use_mad = TRUE</code> , <code>susie</code> is run twice, first by fixing the residual variance to the MAD value, then a second time, initialized to the first fit, but with residual variance estimated the usual way (by maximizing the ELBO). We have found this strategy typically improves reliability of the results by reducing a tendency to converge to poor local optima of the ELBO.
<code>...</code>	Other arguments passed to susie .

Details

This implementation exploits the special structure of X , which means that the matrix-vector product $X^T y$ is fast to compute; in particular, the computation time is $O(n)$ rather than $O(n^2)$ if X were formed explicitly. For implementation details, see the "Implementation of SuSiE trend filtering" vignette by running `vignette("trendfiltering_derivations")`.

Value

A "susie" fit; see [susie](#) for details.

References

R. J. Tibshirani (2014). Adaptive piecewise polynomial estimation via trend filtering. *Annals of Statistics* **42**, 285-323.

Examples

```
set.seed(1)
mu <- c(rep(0, 50), rep(1, 50), rep(3, 50), rep(-2, 50), rep(0, 200))
y <- mu + rnorm(400)
s <- susie_trendfilter(y)
plot(y)
lines(mu, col = 1, lwd = 3)
lines(predict(s), col = 2, lwd = 2)
```

```
# Calculate credible sets (indices of y that occur just before
# changepoints).
susie_get_cs(s)

# Plot with credible sets for changepoints.
susie_plot_changepoint(s, y)
```

univar.order

Ordering of Predictors from Univariate Regression

Description

This function extracts the ordering of the predictors according to the coefficients estimated in a basic univariate regression; in particular, the predictors are ordered in decreasing order by magnitude of the univariate regression coefficient estimate.

Usage

```
univar.order(X, y)
```

Arguments

X	An input design matrix. This may be centered and/or standardized prior to calling function.
y	A vector of response variables.

Value

An ordering of the predictors.

Examples

```
### generate synthetic data
set.seed(1)
n      = 200
p      = 300
X      = matrix(rnorm(n*p),n,p)
beta   = double(p)
beta[1:10] = 1:10
y      = X %%% beta + rnorm(n)

univ.order = univar.order(X,y)
```

univariate_regression *Perform Univariate Linear Regression Separately for Columns of X*

Description

This function performs the univariate linear regression $y \sim x$ separately for each column x of X . The estimated effect size and standard error for each variable are outputted.

Usage

```
univariate_regression(
  X,
  y,
  Z = NULL,
  center = TRUE,
  scale = FALSE,
  return_residuals = FALSE,
  method = c("lmfit", "sumstats")
)

calc_z(X, Y, center = FALSE, scale = FALSE)
```

Arguments

<code>X</code>	<code>n</code> by <code>p</code> matrix of regressors.
<code>y</code>	<code>n</code> -vector of response variables.
<code>Z</code>	Optional <code>n</code> by <code>k</code> matrix of covariates to be included in all regressions. If <code>Z</code> is not <code>NULL</code> , the linear effects of covariates are removed from <code>y</code> first, and the resulting residuals are used in place of <code>y</code> .
<code>center</code>	If <code>center = TRUE</code> , center <code>X</code> , <code>y</code> and <code>Z</code> .
<code>scale</code>	If <code>scale = TRUE</code> , scale <code>X</code> , <code>y</code> and <code>Z</code> .
<code>return_residuals</code>	Whether or not to output the residuals if <code>Z</code> is not <code>NULL</code> .
<code>method</code>	Either “sumstats” (faster implementation) or “lmfit” (uses <code>.lm.fit</code>).
<code>Y</code>	For <code>calc_z</code> , an <code>n</code> -vector or <code>n</code> by <code>T</code> matrix of response variables.

Value

A list with two vectors containing the least-squares estimates of the coefficients (`betahat`) and their standard errors (`sebetahat`). Optionally, and only when a matrix of covariates `Z` is provided, a third vector `residuals` containing the residuals is returned.

Examples

```
set.seed(1)
n = 1000
p = 1000
beta = rep(0,p)
beta[1:4] = 1
X = matrix(rnorm(n*p),nrow = n,ncol = p)
X = scale(X,center = TRUE,scale = TRUE)
y = drop(X %*% beta + rnorm(n))
res = univariate_regression(X,y)
plot(res$betahat/res$sebetahat)
```

unmappable_data	<i>Simulated Fine-mapping Data with Sparse, Oligogenic and Polygenic Effects.</i>
-----------------	---

Description

A simulated data set with 1,000 individuals and 5,000 variants, combining 3 sparse, 5 oligogenic and 15 polygenic non-zero effects. The response is generated under a linear regression model. This data set illustrates fine-mapping with SuSiE-ash and SuSiE-inf.

Format

- unmappable_data is a list with the following elements:
- X** Centered and scaled genotype matrix.
 - y** Simulated response.
 - beta** Simulated effect sizes.
 - h2g** Total proportion of variance in the response explained by the simulated effects.

See Also

The “Fine-mapping with SuSiE-ash and SuSiE-inf” vignette.

Examples

```
data(unmappable_data)
```

Index

- * **data**
 - data_small, 8
 - FinemappingConvergence, 9
 - N2finemapping, 19
 - N3finemapping, 20
 - rss_mismatch_cases, 24
 - SummaryConsistency, 29
 - unmappable_data, 68
- .lm.fit, 67
- absolute.order, 3
- block_coordinate_ascent, 4
- calc_z (univariate_regression), 67
- coef.mr.ash, 5
- coef.susie, 5, 59
- compute_marginal_bhat_shat, 6
- compute_suff_stat, 7
- data_small, 8
- estimate_s_rss, 8
- FinemappingConvergence, 9
- get.full.posterior, 10, 16
- get_cs_correlation, 11
- kriging_rss, 12
- mr.ash, 10, 13
- mr.ash.rss, 17
- N2finemapping, 19
- N3finemapping, 20
- path.order, 21
- plot, 42
- predict.mr.ash, 16, 22
- predict.susie, 23
- print.summary.susie (summary.susie), 27
- print.summary.susie_post_outcome_configuration, 23
- rss_mismatch_cases, 24
- slot_prior_betabinom, 25, 34, 52, 58, 64
- slot_prior_poisson, 34, 52, 58, 64
- slot_prior_poisson (slot_prior_betabinom), 25
- summary.susie, 27
- summary.susie_post_outcome_configuration, 27
- SummaryConsistency, 29
- susie, 11, 29, 36, 38, 41, 52, 65
- susie_auto, 35
- susie_get_cs, 29, 33, 34, 50, 51, 57–59, 63, 64
- susie_get_cs (susie_get_objective), 37
- susie_get_cs_attainable, 59
- susie_get_cs_attainable (susie_get_objective), 37
- susie_get_lfsr (susie_get_objective), 37
- susie_get_niter (susie_get_objective), 37
- susie_get_objective, 37
- susie_get_pip, 59
- susie_get_pip (susie_get_objective), 37
- susie_get_posterior_mean (susie_get_objective), 37
- susie_get_posterior_samples (susie_get_objective), 37
- susie_get_posterior_sd (susie_get_objective), 37
- susie_get_prior_variance (susie_get_objective), 37
- susie_get_residual_variance (susie_get_objective), 37
- susie_init_coef, 40
- susie_plot, 41
- susie_plot_changepoint, 42, 43

`susie_plot_iteration(susie_plot)`, [41](#)
`susie_post_outcome_configuration`, [44](#)
`susie_rss`, [47](#), [54](#), [55](#), [58](#), [59](#)
`susie_rss_control`, [52](#), [53](#)
`susie_rss_lambda`, [47](#), [54](#)
`susie_ser`, [58](#)
`susie_ss`, [60](#)
`susie_trendfilter`, [64](#)

`tempdir`, [42](#)

`univar.order`, [66](#)
`univariate_regression`, [67](#)
`unmappable_data`, [68](#)