

# Package: dependentsimr (via r-universe)

June 3, 2026

**Title** Simulate Omics-Scale Data with Dependency

**Version** 1.0.0.0

**Description** Using a Gaussian copula approach, this package generates simulated data mimicking a target real dataset. It supports normal, Poisson, empirical, and 'DESeq2' (negative binomial with size factors) marginal distributions. It uses an low-rank plus diagonal covariance matrix to efficiently generate omics-scale data. Methods are described in: Yang, Grant, and Brooks (2025) <[doi:10.1101/2025.01.31.634335](https://doi.org/10.1101/2025.01.31.634335)>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Imports** rlang (>= 1.0.0)

**Suggests** DESeq2 (>= 1.40.0), S4Vectors (>= 0.44.0), SummarizedExperiment (>= 1.36.0), MASS (>= 7.3), corpcor (>= 1.6.0), testthat (>= 3.0.0), Matrix (>= 1.7), sparsesvd (>= 0.2), knitr (>= 1.50), rmarkdown, BiocManager, remotes, tidyverse (>= 2.0.0)

**Depends** R (>= 4.2)

**LazyData** true

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**Repository** <https://tgbrooks.r-universe.dev>

**Date/Publication** 2025-10-06 18:06:45 UTC

**RemoteUrl** <https://github.com/tgbrooks/dependentsimr>

**RemoteRef** HEAD

**RemoteSha** 1af2c556bff0e51722e5c0a52117df2dd4bb4000

## Contents

draw_from_multivariate_corr . . . . .	2
get_random_structure . . . . .	3
match_with_spiked_wishart . . . . .	4
multi_sample_spiked_wishart . . . . .	5
read_counts . . . . .	6
remove_dependence . . . . .	7
sample_spiked_wishart . . . . .	7
sample_spiked_wishart_and_jac . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

draw\_from\_multivariate\_corr

*Draw random samples from the given random structure*

---

### Description

Draw random samples from the given random structure

### Usage

```
draw_from_multivariate_corr(random_structure, n_samples, size_factors = NULL)
```

### Arguments

`random_structure` The structure of the random data to draw, computed by `get_random_structure()`

`n_samples` The number of samples to generate

`size_factors` Vector of length `n_samples` of size factors for DESeq2-type data. By default, all samples are given 1. Not used if no datasets has type DESeq2.

### Value

A list of generated datasets. List names correspond to those of the datasets used to generate the random structure. Each dataset has `n_samples` columns.

### Examples

```
## # Generate example data with Sigma as its covariance matrix
library(MASS)
Sigma = matrix(c(
  1,    0.8,  0,  0,  0,  0,
  0.8,  1,    0,  0,  0,  0,
  0,    0,    1,  0,  0,  0,
  0,    0,    0,  1,  0,  0,
  0,    0,    0,  0,  1,  0.3,
  0,    0,    0,  0,  0.3,  1
), 6, 6)
```

```
), nrow=6, ncol=6)
norm_data <- t(mvrnorm(n=20, mu=c(0,0,0,0,0,0), Sigma=Sigma))

# Simulate draws mimicking that data
rs_normal <- get_random_structure(list(data=norm_data), method="pca", rank=2, type="normal")
draws_normal <- draw_from_multivariate_corr(rs_normal, n_samples=30)
```

---

get\_random\_structure *Compute structure of dependency from a given data*

---

## Description

Compute structure of dependency from a given data

## Usage

```
get_random_structure(datasets, method, rank = 2, types = "normal")
```

## Arguments

datasets	A list of data matrices that we will mimic. All datasets have samples in the columns and features (e.g., genes, proteins) in the rows. All datasets must have the same samples in corresponding columns.
method	One of 'pca', 'spiked Wishart', or 'corp.cor', for the method of determining a dependency structure
rank	Number of PCA components to approximate the dependence structure of. Only used if method = 'pca'.
types	The marginal distribution types ('normal', 'poisson', 'DESeq2', or 'empirical'), as a list with entries corresponding to datasets. If just a single value is provided, then it is used for all datasets.

## Value

A random structure element suitable for use with draw\_from\_multivariate\_corr().

## Examples

```
# Simple data to mimic
true_means <- c(0, 1, 10, 1000)
data <- rpois(4*12, true_means) |> matrix(4, 12)

# Simulate draws mimicking that data
rs <- get_random_structure(list(data=data), method="pca", rank=2, type="poisson")
draws <- draw_from_multivariate_corr(rs, n_samples=30)
```

---

`match_with_spiked_wishart`

*Compute what spiked SD values will give you the desired top eigenvalues by iteratively solving*

---

## Description

Compute what spiked SD values will give you the desired top eigenvalues by iteratively solving

## Usage

```
match_with_spiked_wishart(  
  desired_eigenvalues,  
  rank,  
  num_observations,  
  num_variables,  
  population_sd = 1,  
  num_iterations = 20,  
  num_samples_per_iter = 300  
)
```

## Arguments

<code>desired_eigenvalues</code>	Vector of top eigenvalues of sample covariance matrix that should be approximated
<code>rank</code>	number of 'spiked' dimensions to fit
<code>num_observations</code>	Number of observations (samples, columns) in the data matrix
<code>num_variables</code>	Number of variables (features, rows) in the data matrix
<code>population_sd</code>	Standard deviation for all non-spiked dimensions
<code>num_iterations</code>	Number of iterations to perform to optimize. Increase to get a closer fit
<code>num_samples_per_iter</code>	Number of eigenvalues to samples to estimate mean eigenvalues for each iteration. Increase to get a closer fit

## Value

Fit values of spiked SDs that approximately match the desired eigenvalues

## Examples

```
# First, simulate a dataset with known true values to see if we can get close  
true_spiked_sd <- c(500, 100)  
num_variables <- 1000  
num_observations <- 10-1
```

```

sampled_eigenvalues <- sample_spiked_wishart(true_spiked_sd, num_observations,
      num_variables, num_eigs=2)
# Fit some spiked SDs that give eigenvalues similar to sampled_eigenvalues
fit <- match_with_spiked_wishart(
  sampled_eigenvalues,
  rank=2,
  num_observations,
  num_variables,
  num_observations=5,
  num_samples_per_iter=50
)
# fit$spiked_sd should now be close to giving the specified sampled_eigenvalues (in expectation)
# fit$spiked_sd won't match the original true_spiked_sd too closely since it only
# fits the single sample # that gave sampled_eigenvalue
# fit$population_sd gives fit value for the population SD

```

---

```
multi_sample_spiked_wishart
```

*Compute means of each singular value and the mean Jacobian, see  
sample\_spiked\_wishart\_and\_jac*

---

## Description

Compute means of each singular value and the mean Jacobian, see `sample_spiked_wishart_and_jac`

## Usage

```

multi_sample_spiked_wishart(
  count,
  spiked_sd,
  num_observations,
  num_variables,
  population_sd = 1,
  num_eigs = 0
)

```

## Arguments

<code>count</code>	The number of samples to compute the mean of
<code>spiked_sd</code>	The spiked standard deviations
<code>num_observations</code>	The number of observations (aka samples or columns)
<code>num_variables</code>	The number of variables (aka features or rows)
<code>population_sd</code>	the standard deviation of all non-spiked components ( <code>num_variables - length(spiked_sd)</code> of them)
<code>num_eigs</code>	The number of eigenvalues to compute. If 0 compute all of them using dense matrix routines. If greater than zero, use sparse matrices and compute that many top eigenvalues.

**Value**

List with a vector of mean singular values of  $G$  where  $G$  is a random  $\text{num\_variables} \times \text{num\_observations}$  matrix with iid columns from  $N(0, \text{Sigma})$  where  $\text{Sigma}$  is diagonal with entries  $\text{spiked\_sd}^2$  and all the remaining are  $\text{population\_sd}^2$ . and also the mean Jacobian, where  $\partial G_{ij} / \partial \text{spiked\_sd}_j$  is the derivative of the  $i$ th singular value with respect to the  $j$ th spiked SD, and the gradient of the  $\text{population\_sd}$  parameter

**Examples**

```
# Sample 10 times from the spiked Wishart distribution with (500, 100, 1, ..., 1) singular values
# and take the means of the singular values as well as derivatives (jacobian and pop_sd_grad)
mean_vals <- multi_sample_spiked_wishart(
  count = 10,
  spiked_sd = c(500, 100),
  num_observations = 10-1,
  num_variables = 1000,
  num_eigs = 3
)
mean_vals$singular_vals
mean_vals$jacobian
mean_vals$pop_sd_grad
```

---

read\_counts

*GSE151923: cortex from 6-month-old wildtype C57BL/6 mice*


---

**Description**

Quantifications of bulk RNA-seq data of liver samples from 12 male mice. This includes only a subset of mice and genes from the experiment, for size.

**Usage**

```
read_counts
```

**Format**

```
read_counts:
```

A tibble with 1,000 rows and 13 columns. Each column is one mouse sample from GSE151923. Each row is a ENSEMBL mouse gene/transcript with the quantified values for each sample. Gene identifiers are in the first column.

**Source**

<https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE151923>

---

remove_dependence	<i>Remove all dependence in a random structure</i>
-------------------	--

---

**Description**

Remove all dependence in a random structure

**Usage**

```
remove_dependence(random_structure)
```

**Arguments**

random\_structure  
A random structure from `get_random_structure()`

**Value**

The random structure with dependency removed, so all data generated from it will be independent.

**Examples**

```
library(MASS)
Sigma = matrix(c(
  1,    0.8,  0,  0,  0,  0,
  0.8,  1,    0,  0,  0,  0,
  0,    0,    1,  0,  0,  0,
  0,    0,    0,  1,  0,  0,
  0,    0,    0,  0,  1,  0.3,
  0,    0,    0,  0,  0.3, 1
), nrow=6, ncol=6)
norm_data <- t(mvrnorm(n=20, mu=c(0,0,0,0,0,0), Sigma=Sigma))

# Simulate draws mimicking that data but without any dependence
rs_normal <- get_random_structure(list(data=norm_data), method="pca", rank=2, type="normal")
rs_indep <- remove_dependence(rs_normal)
draws_indep <- draw_from_multivariate_corr(rs_indep, n_samples=30)
```

---

sample_spiked_wishart	<i>Efficiently sample the singular values corresponding to a random Wishart matrix with spiked eigenvalues Specifically, if <math>W = G G^T</math> with each column of <math>G</math> drawn iid from <math>N(0, \text{Sigma})</math>, then <math>W</math> is a Wishart matrix and this function samples the singular values of <math>G</math>. The eigenvalues of <math>W</math> are just the squares of the singular values. Here, <math>\text{Sigma}</math> is diagonal with its leading entries from <code>spiked_sd^2</code> and all remaining entries are <code>population_sd^2</code>.</i>
-----------------------	--

---

**Description**

Efficiently sample the singular values corresponding to a random Wishart matrix with spiked eigenvalues. Specifically, if  $W = G G^T$  with each column of  $G$  drawn iid from  $N(0, \Sigma)$ , then  $W$  is a Wishart matrix and this function samples the singular values of  $G$ . The eigenvalues of  $W$  are just the squares of the singular values. Here,  $\Sigma$  is diagonal with its leading entries from `spiked_sd^2` and all remaining entries are `population_sd^2`.

**Usage**

```
sample_spiked_wishart(
  spiked_sd,
  num_observations,
  num_variables,
  population_sd = 1,
  num_eigs = 0
)
```

**Arguments**

<code>spiked_sd</code>	The spiked standard deviations
<code>num_observations</code>	The number of observations (aka samples or columns)
<code>num_variables</code>	The number of variables (aka features or rows)
<code>population_sd</code>	the standard deviation of all non-spiked components ( <code>num_variables - length(spiked_sd)</code> of them)
<code>num_eigs</code>	The number of eigenvalues to compute. If 0 compute all of them using dense matrix routines. If greater than zero, use sparse matrices and compute that many top eigenvalues.

**Value**

Vector of random singular values of  $G$  where  $G$  is a random `num_variables` x `num_observations` matrix with iid columns from  $N(0, \Sigma)$  where  $\Sigma$  is diagonal with entries `spiked_sd^2` and all the remaining are `population_sd^2`.

**Examples**

```
set.seed(0)
# Sample eigenvalues of a covariance matrix of a 10 sample study with 1000 variables such that
# the top two (underlying true distribution of the data, not sample) principal components
# have SDs of 100 and 10 and the remaining 98 have 1
sample_spiked_wishart(
  spiked_sd = c(500, 100),
  num_variables = 1000,
  num_observations = 10-1,
  num_eigs = 3
)
```

---

sample\_spiked\_wishart\_and\_jac

*Efficiently sample the singular values corresponding to a random Wishart matrix with spiked eigenvalues and the Jacobian I.e., these are the singular values of  $G$  if  $GG^T$  is Wishart. The square of these give the eigenvalues of the random Wishart matrix.*

---

### Description

Efficiently sample the singular values corresponding to a random Wishart matrix with spiked eigenvalues and the Jacobian I.e., these are the singular values of  $G$  if  $GG^T$  is Wishart. The square of these give the eigenvalues of the random Wishart matrix.

### Usage

```
sample_spiked_wishart_and_jac(
  spiked_sd,
  num_observations,
  num_variables,
  population_sd = 1,
  num_eigs = 0
)
```

### Arguments

spiked_sd	The spiked standard deviations
num_observations	The number of observations (aka samples or columns)
num_variables	The number of variables (aka features or rows)
population_sd	the standard deviation of all non-spiked components ( $\text{num\_variables} - \text{length}(\text{spiked\_sd})$ of them)
num_eigs	The number of eigenvalues to compute. If 0 compute all of them using dense matrix routines. If greater than zero, use sparse matrices and compute that many top eigenvalues.

### Value

List with a vector of random singular values of  $G$  where  $G$  is a random  $\text{num\_variables} \times \text{num\_observations}$  matrix with iid columns from  $N(0, \text{Sigma})$  where  $\text{Sigma}$  is diagonal with entries  $\text{spiked\_sd}^2$  and all the remaining are  $\text{population\_sd}^2$ . and also the Jacobian, where  $\partial G_{ij} / \partial \text{spiked\_sd}_i$  is the derivative of the  $i$ th singular value with respect to the  $j$ th spiked SD and the gradient of the  $\text{population\_sd}$  variable

**Examples**

```
set.seed(0)
# Sample eigenvalues of a covariance matrix of a 10 sample study with 1000 variables such that
# the top two (underlying true distribution of the data, not sample) principal components
# have SDs of 100 and 10 and the remaining 98 have 1
res = sample_spiked_wishart_and_jac(
  spiked_sd = c(500, 100),
  num_variables = 1000,
  num_observations = 10-1,
  num_eigs = 3
)
res$singular_vals # singular values of G, (i.e., square roots of the eigenvalues of W = G G^T)
res$jacobian # jacobian of the singular values with respect to the spiked_sd's
res$pop_sd_grad # gradient of population_sd parameter
```

# Index

## \* datasets

- read\_counts, [6](#)
- draw\_from\_multivariate\_corr, [2](#)
- get\_random\_structure, [3](#)
- match\_with\_spiked\_wishart, [4](#)
- multi\_sample\_spiked\_wishart, [5](#)
- read\_counts, [6](#)
- remove\_dependence, [7](#)
- sample\_spiked\_wishart, [7](#)
- sample\_spiked\_wishart\_and\_jac, [9](#)