Package 'landgraph'

July 6, 2026

Type Package

Title Graphs and Covariance for Landscape Genetics

Version 0.0.1

Date 2026-06-11

Description Shared, dependency-light primitives for landscape-genetic network methods: a lightweight deme/landscape graph (vertex coordinates and an undirected edge list) with constructors from coordinates; genetic covariance and distance from biallelic or multiallelic data (the Yang-style normalized-dosage covariance and the Dyer-style multivariate covariance); and antisymmetric per-edge directional covariate builders (the gradient of a scalar potential, and the projection of a vector flow field). Used by 'terradish' (symmetric resistance) and 'dragonflow' (asymmetric gene flow). No compiled code.

License BSD_3_clause + file LICENSE

Encoding UTF-8

Depends R (>= 4.0.0)

Imports stats

Suggests terra, deldir, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

URL https://github.com/wpeterman/landgraph

BugReports https://github.com/wpeterman/landgraph/issues

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Bill Peterman [aut, cre, cph],
Nathaniel S. Pope [aut, cph]

Maintainer Bill Peterman <peterman.73@osu.edu>

Repository CRAN

Date/Publication 2026-07-06 13:50:11 UTC

Contents

`cov_from_biallelic` *Genetic covariance from biallelic allele counts*

Description

Computes a sample covariance matrix from counts of the derived allele across biallelic (SNP) markers. Rows may be individuals, populations, or any other sampled genetic unit. The result is directly suitable as the response `S` in `wishart_covariance`.

Usage

```
cov_from_biallelic(
  Y,
  N = NULL,
  ploidy = 2,
  monomorphic = c("drop", "error"),
  tol = sqrt(.Machine$double.eps)
)
```

Arguments

`Y`	Numeric matrix of derived-allele counts with sampled units in rows and loci in columns. For diploid individuals genotyped as 0/1/2, this is the standard dosage matrix.
`N`	Optional haploid sample-size specification. Controls how many chromosomes were sampled at each locus for each unit: • `NULL`: use `ploidy` for all cells. • A scalar: recycled to all cells. • A vector of length `nrow(Y)`: row-specific sample sizes (same at every locus for that row). • A vector of length `ncol(Y)`: locus-specific sample sizes (same across all rows). • A matrix matching `dim(Y)`: fully general cell-level sizes.

`ploidy`	Haploid chromosome count per unit, used when `N = NULL`. The default `2` corresponds to diploid individuals with dosage coding 0/1/2 (so each individual contributes two chromosomes at each locus).
`monomorphic`	How to handle loci with pooled allele frequency 0 or 1 (fixed across all rows). `"drop"` (default) removes them and emits a warning; `"error"` stops with an error to preserve historical strict behavior.
`tol`	Tolerance for identifying monomorphic loci. A locus is treated as monomorphic when its pooled frequency is within `tol` of 0 or 1.

Details

Let $p_\ell = \sum_i Y_{i\ell} / \sum_i N_{i\ell}$ be the pooled derived-allele frequency at locus ℓ. The normalized allele dosage is:

$$Z_{i\ell} = \frac{Y_{i\ell} - N_{i\ell} p_\ell}{\sqrt{N_{i\ell} p_\ell (1 - p_\ell)}}$$

The function returns $ZZ^\top / L$, where L is the number of retained (polymorphic) loci. The resulting matrix is the sample covariance in normalized allele-frequency space, analogous to the genomic relationship matrix of Yang et al. (2010).

To convert the covariance to a pairwise distance matrix, use `dist_from_cov`. For non-biallelic or multiallelic data, use `cov_from_genetic_data`.

Value

A symmetric positive-semidefinite numeric matrix of pairwise genetic covariances. Row and column names match those of `Y` when present.

References

Yang J, Benyamin B, McEvoy BP, Gordon S, Henders AK, Nyholt DR, Madden PA, Heath AC, Martin NG, Montgomery GW, Goddard ME, Visscher PM. 2010. Common SNPs explain a large proportion of the heritability for human height. Nature Genetics 42(7):565-569. doi:10.1038/ng.608

See Also

`cov_from_genetic_data`, `fst_from_biallelic`, `dist_from_biallelic`, `dist_from_cov`, `wishart_covariance`

Examples

```
# Individual diploid dosage matrix: 3 individuals, 3 SNPs
Y <- matrix(c(2, 1, 0,
              1, 1, 1,
              0, 1, 2), nrow = 3, byrow = TRUE)
cov_from_biallelic(Y)          # ploidy = 2 by default

# Population-level counts with unequal sampling
Y2 <- matrix(c(10, 4, 1,
               5,  5, 3), nrow = 2, byrow = TRUE)
```

```
N2 <- matrix(c(20, 20, 10,
               10, 10, 6), nrow = 2, byrow = TRUE)
cov_from_biallelic(Y2, N = N2)
```

`cov_from_genetic_data` *Covariance from multivariate genetic data*

Description

Calculates individual- or population-level genetic covariance from multivariate genetic data. This is useful for microsatellite, multiallelic, SNP dosage, PCA score, or other numeric encodings.

Usage

```
cov_from_genetic_data(
  x,
  groups = NULL,
  input = c("features", "allele_calls"),
  loci = NULL,
  center = TRUE,
  scale = TRUE,
  tol = sqrt(.Machine$double.eps),
  diagonal = c("auto", "within", "gower"),
  normalize = c("none", "features")
)
```

Arguments

`x`	Genetic data. For `input = "features"`, a numeric matrix or data frame with individuals in rows and genetic features in columns. For `input = "allele_calls"`, a matrix or data frame of allele calls with one column per allele copy.
`groups`	Optional factor, character, or integer vector assigning each row of `x` to a population. If `NULL`, each row is treated as an individual sampled unit and an individual-level covariance matrix is returned.
`input`	Input format. `"features"` treats `x` as an already numeric feature matrix. `"allele_calls"` converts allele calls to per-locus allele dosage columns before calculating covariance.
`loci`	Required for `input = "allele_calls"`; a vector with one entry per column of `x` identifying the locus for each allele-copy column. For a diploid microsatellite matrix with two columns per locus, the two columns for each locus should have the same `loci` value.
`center`	Should feature columns be centered by their global mean before covariance calculation? Default `TRUE`.
`scale`	Should feature columns be divided by their global standard deviation? Default `TRUE`. Constant features are dropped.

tol	Tolerance used to identify constant features when scale = TRUE.
diagonal	How to set the returned covariance diagonal. "auto" uses "within" for grouped population data with replication and "gower" for individual-level data. "within" replaces the Gower covariance diagonal with the within-population genetic variance, following the population-graph covariance construction. "gower" leaves the double-centered distance diagonal unchanged.
normalize	Should covariance scale be normalized? "none" returns sums over retained features. "features" divides the covariance, squared distances, and within-population variances by the number of retained features.

Details

The function first obtains a numeric feature matrix. Microsatellite data can be supplied as allele calls by setting input = "allele_calls"; these are converted to allele dosage columns, one column per locus-allele combination. Missing allele calls are imputed to the modal observed allele within each locus, with a message reporting the number of imputed calls.

For Wishart models, the nu argument is the effective degrees of freedom in the empirical covariance. With biallelic SNPs, each retained polymorphic SNP contributes one independent standardized allele-frequency column, so nu is usually the retained SNP count (reduced for linkage disequilibrium). With microsatellites, use the number of loci L as the conservative default: allele frequencies within a locus are correlated (they sum to a constant), so the locus is the natural unit of information in population genetics. $\sum_l (K_l - 1)$, where K_l is the number of observed alleles at locus l, is an upper bound that assumes within-locus alleles are independent; this assumption is not generally satisfied, and using it can yield over-confident standard errors and model-selection statistics. The true effective ν for microsatellite data lies somewhere between L and $\sum_l (K_l - 1)$. Report the value used and conduct a sensitivity analysis across that range. See wishart_covariance for the full explanation.

If groups = NULL, rows of the processed feature matrix are used directly and squared Euclidean distances among individuals are transformed to a covariance matrix by Gower double-centering. This produces a positive semidefinite row-level covariance matrix up to numerical tolerance.

If groups is supplied, population centroids are calculated in feature space before the Gower transform. With diagonal = "within", the diagonal is replaced by the sum of within-population feature variances. This matches the covariance construction used before population-graph partial-correlation filtering in Dyer-style population graph workflows.

Missing values are not currently supported because common microsatellite imputation choices can change the resulting covariance.

If the grouped Dyer-style result is used as S in wishart_covariance, inspect the eigenvalues first. The within-population diagonal is useful for population-graph workflows, but it does not guarantee a positive-definite covariance matrix for every dataset.

Value

A symmetric numeric matrix with one row and column per individual (when groups = NULL) or population (when groups is supplied). The matrix is positive semi-definite up to numerical tolerance, except when diagonal = "within" is used (see Details). The following attributes are attached:

centroids Feature matrix of population centroids (or individual feature vectors when ungrouped).

`centroid_distance2` Squared Euclidean distance matrix among centroids.

`within_variance` Named vector of within-population variances (only when `groups` is supplied and some populations have ≥ 2 individuals).

`unit_size` Named integer vector of group sizes.

`feature_center`, `feature_scale` Centering and scaling constants applied to each retained feature.

`retained_features` Names of the features kept after constant features were dropped.

`level` Either `"individual"` or `"population"`.

`diagonal` The `diagonal` method actually used.

`normalizer` Divisor applied to the covariance and distances (1 for `normalize = "none"`, number of retained features for `normalize = "features"`).

References

Gower JC. 1966. Some distance properties of latent root and vector methods used in multivariate analysis. Biometrika 53(3-4):325-338. doi:10.1093/biomet/53.34.325

Dyer RJ, Nason JD. 2004. Population graphs: the graph theoretic shape of genetic structure. Molecular Ecology 13(7):1713-1727. doi:10.1111/j.1365294X.2004.02177.x

Dyer RJ. 2015. Population graphs and landscape genetics. Annual Review of Ecology, Evolution, and Systematics 46:327-342. doi:10.1146/annurevecolsys112414054150

See Also

`wishart_covariance`, `cov_from_biallelic`, `dist_from_cov`, `simulate_covariance_response`

Examples

```
# Numeric allele dosage / feature matrix
x <- matrix(c(0, 1,
              1, 1,
              2, 0,
              2, 1,
              0, 2,
              1, 2),
            ncol = 2, byrow = TRUE)
cov_from_genetic_data(x)

groups <- rep(c("pop1", "pop2", "pop3"), each = 2)
cov_from_genetic_data(x, groups = groups)

# Microsatellite-style allele-call columns: two allele copies per locus
alleles <- data.frame(
  loc1_a = c(100, 100, 102, 102, 104, 104),
  loc1_b = c(100, 102, 102, 104, 104, 100),
  loc2_a = c(200, 202, 200, 202, 204, 204),
  loc2_b = c(202, 202, 204, 204, 204, 200)
)
cov_from_genetic_data(
```

```
  alleles,
  groups = groups,
  input = "allele_calls",
  loci = c("loc1", "loc1", "loc2", "loc2")
)
```

---

deme_graph *Lightweight deme / landscape graph from coordinates*

Description

Builds the minimal spatial graph that DRAGON-style network methods consume: a set of vertex coordinates and an undirected edge list. For deme-scale problems (dozens to a few hundred nodes) this avoids the heavier raster / conductance graph builders, and the returned object is drop-in compatible with the graph consumed by `dragon` (in dragonflow) and with a `terradish_graph`.

Usage

```
deme_graph(
  coords,
  neighbours = c("delaunay", "knn", "lattice"),
  k = 6L,
  queen = FALSE
)
```

Arguments

`coords`	A two-column numeric matrix or data frame of node (deme) coordinates, one row per node (columns x, y).
`neighbours`	Edge construction. `"delaunay"` (the Delaunay triangulation; needs the **deldir** package), `"knn"` (symmetric k-nearest-neighbour adjacency), or `"lattice"` (rook or queen adjacency for points lying on an integer grid).
`k`	Number of neighbours for `neighbours = "knn"`.
`queen`	For `neighbours = "lattice"`, also connect diagonal (queen) neighbours; the default rook adjacency reproduces `conductance_surface(directions = 4)`.

Value

An object of class `c("landgraph", "terradish_graph")` with `vertex_coordinates` (an `n x 2` matrix), `edge_pairs` (an `m x 2` integer matrix of 1-based undirected edges, each pair once, `a < b`), and `n_vertices`.

See Also

`edge_gradient`, `edge_flow`

Examples

```
coords <- as.matrix(expand.grid(x = 0:4, y = 0:4))
g <- deme_graph(coords, neighbours = "lattice")
nrow(g$edge_pairs)
```

`dist_from_biallelic` *Genetic distance matrix from biallelic allele counts*

Description

Computes pairwise squared genetic distances from biallelic (SNP) allele counts. This is a convenience wrapper for `dist_from_cov(cov_from_biallelic(Y, N))`.

Usage

```
dist_from_biallelic(Y, N)
```

Arguments

`Y`	Numeric matrix of derived-allele counts (populations/individuals in rows, loci in columns).
`N`	Numeric matrix of haploid sample sizes with the same dimensions as `Y`. See `cov_from_biallelic` for flexible `N` specifications.

Value

A symmetric numeric matrix of pairwise squared distances with zero diagonal. Suitable for use with `mlpe` or `generalized_wishart`.

See Also

`cov_from_biallelic`, `dist_from_cov`, `fst_from_biallelic`

Examples

```
Y <- matrix(c(2, 1, 0,
              1, 1, 1,
              0, 1, 2), nrow = 3, byrow = TRUE)
N <- matrix(2, nrow = 3, ncol = 3)
dist_from_biallelic(Y, N)
```

dist_from_cov *Squared-distance matrix from a covariance matrix*

Description

Converts a covariance matrix to its associated squared pairwise distance matrix. This inverts the Gower double-centering used by cov_from_genetic_data.

Usage

```
dist_from_cov(Cov)
```

Arguments

Cov	A square numeric covariance matrix. Does not need to be full-rank (e.g. the output of cov_from_biallelic is positive semi-definite).

Details

The squared distance between units i and j is:

$$D_{ij} = C_{ii} + C_{jj} - 2C_{ij}$$

which is the law-of-cosines identity relating a covariance matrix to its implied squared Euclidean distances. The diagonal of the returned matrix is zero.

Use this function to convert a covariance matrix produced by cov_from_biallelic or cov_from_genetic_data into a distance matrix suitable for generalized_wishart or mlpe.

Value

A symmetric numeric matrix of the same dimension as Cov with zero diagonal and non-negative off-diagonal entries.

See Also

cov_from_biallelic, cov_from_genetic_data, dist_from_biallelic

Examples

```
Cov <- matrix(c(2.0, 1.2, 0.8,
                1.2, 1.5, 0.7,
                0.8, 0.7, 1.1), nrow = 3, byrow = TRUE)
dist_from_cov(Cov)

# Round-trip: covariance -> distance -> (verify non-negative, zero diag)
Y <- matrix(c(2, 1, 0, 1, 1, 1, 0, 1, 2), nrow = 3, byrow = TRUE)
S_cov  <- cov_from_biallelic(Y)
S_dist <- dist_from_cov(S_cov)
```

```
diag(S_dist)   # all zeros
```

---

edge_flow *Circulation (flow-field) edge covariate from a spatial vector field*

Description

Builds an antisymmetric per-edge covariate from a spatial vector field (for example wind or current), for the `circulation` argument of `dragon`. `edge_gradient` takes the gradient of a scalar potential, which is curl-free and yields a reversible generator whose stationary distribution is collinear with the potential. A vector flow field can carry a rotational/curl component instead, making the directed generator non-reversible and its stationary distribution non-collinear with any scalar covariate.

Usage

```
edge_flow(field, data)
```

Arguments

`field`	The vector field at the active graph cells: a two-column numeric matrix or data frame (x- and y-components, in vertex order), a two-layer `terra::SpatRaster` (sampled at the vertex coordinates), or a function of a two-column coordinate matrix returning such a two-column field.
`data`	A `terradish_graph` from `deme_graph` (must carry `vertex_coordinates`).

Details

For undirected edge (a, b) the covariate is the mean field along the edge projected onto the edge direction, $c_{ab} = \frac{1}{2}(f_a + f_b) \cdot (xy_b - xy_a)$; it is antisymmetric by construction, and `dragon` applies $-c_{ab}$ to the reverse edge $b \to a$.

Value

A numeric vector with one entry per undirected edge in `data$edge_pairs`, ready to pass as `dragon(circulation = )`.

See Also

`edge_gradient`, `dragon`

Examples

```
coords <- as.matrix(expand.grid(x = 0:2, y = 0:2))
g <- deme_graph(coords, neighbours = "lattice")
field <- matrix(c(1, 0), nrow = nrow(coords), ncol = 2, byrow = TRUE)
edge_flow(field, g)
```

edge_gradient *Directional edge covariates from a spatial layer*

Description

Builds the antisymmetric per-directed-edge covariate $d_{ab} = x_a - x_b$ used by `terradish_directed` to drive directional gene flow (e.g. elevation drop, flow accumulation, wind potential). $d_{ab} = -d_{ba}$, so a positive coefficient makes movement from high to low `x` faster.

Usage

```
edge_gradient(x, data)
```

Arguments

`x`	A `terra::SpatRaster` layer (or a numeric vector with one value per active graph cell) giving the potential whose gradient drives direction.
`data`	A `terradish_graph` from `deme_graph`.

Value

A list with `edges` (an integer matrix of directed edges, columns `a`, `b`, 1-based node indices) and `d` (the matching numeric vector $d_{ab} = x_a - x_b$).

See Also

`terradish_directed`

Examples

```
coords <- as.matrix(expand.grid(x = 0:2, y = 0:2))
g <- deme_graph(coords, neighbours = "lattice")
elevation <- c(10, 20, 30, 25, 15, 5, 8, 18, 28)
edge_gradient(elevation, g)
```

fst_from_biallelic *F_ST from biallelic allele counts*

Description

Estimates pairwise F_{ST} from counts of the derived allele across biallelic markers using the ratio-of-averages estimator of Bhatia et al. (2013).

Usage

```
fst_from_biallelic(Y, N)
```

Arguments

Y	Numeric matrix of derived-allele counts with rows as populations and columns as loci.
N	Numeric matrix of sampled haploid chromosomes with the same dimensions as Y. For diploid data without missing genotypes, N = 2 * n_individuals for each cell.

Details

The estimator computes, for each locus, the squared allele-frequency difference between populations, corrects for within-population heterozygosity, and then forms a ratio of locus-averages (Bhatia et al. 2013). Diagonal entries are set to 0.

The result is not guaranteed to lie in $[0, 1]$ for every pair; small negative values can arise from sampling noise in very similar populations.

To use F_{ST} as the response in terradish, pass the matrix directly to the formula left-hand side with mlpe or generalized_wishart.

Value

A symmetric numeric matrix of pairwise F_{ST} with zero diagonal.

References

Bhatia G, Patterson N, Sankararaman S, Price AL. 2013. Estimating and interpreting F_{ST}: the impact of rare variants. Genome Research 23(9):1514-1521. doi:10.1101/gr.154831.113

See Also

cov_from_biallelic, dist_from_biallelic

Examples

```
Y <- matrix(c(2, 1, 0,
              1, 1, 1,
              0, 1, 2), nrow = 3, byrow = TRUE)
N <- matrix(2, nrow = 3, ncol = 3)
fst_from_biallelic(Y, N)
```

Index