Package 'dress.graph'

March 23, 2026

Type Package

Title DRESS - A Continuous Framework for Structural Graph Refinement

Version 0.6.2

Description DRESS is a deterministic, parameter-free framework that iteratively refines the structural similarity of edges in a graph to produce a canonical fingerprint: a real-valued edge vector, obtained by converging a non-linear dynamical system to its unique fixed point. The fingerprint is isomorphism-invariant by construction, guaranteed bitwise-equal across any vertex labeling, numerically stable (no overflow, no error amplification, no undefined behavior), fast and embarrassingly parallel to compute:
DRESS total runtime is O(I * m * d_max) for I iterations to convergence, and convergence is guaranteed by Birkhoff contraction.

License MIT + file LICENSE

URL https://github.com/velicast/dress-graph, https://velicast.github.io/dress-graph/

BugReports https://github.com/velicast/dress-graph/issues

Encoding UTF-8

NeedsCompilation yes

SystemRequirements OpenMP, CUDA Toolkit (optional, for GPU acceleration)

Author Eduar Castrillo Velilla [aut, cre] (ORCID: <https://orcid.org/0009-0005-2492-0957>)

Maintainer Eduar Castrillo Velilla <velicast@outlook.com>

Repository CRAN

Date/Publication 2026-03-23 11:40:02 UTC

Contents

---

cuda *CUDA-accelerated DRESS functions.*

Description

An environment containing GPU-accelerated versions of `dress_fit` and `delta_dress_fit` with identical signatures. Switch from CPU to GPU by prefixing calls with `cuda$`.

Details

The cuda environment provides:

`cuda$dress_fit(...)` GPU-accelerated `dress_fit`. Same arguments and return value.

`cuda$delta_dress_fit(...)` GPU-accelerated `delta_dress_fit`. Same arguments and return value.

CUDA support requires rebuilding the package with `DRESS_CUDA=1`. If CUDA is not available, calling either function raises an error.

Examples

```
## Not run:
  # CPU
  r1 <- dress_fit(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L))

  # CUDA -- same signature
  r2 <- cuda$dress_fit(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L))

## End(Not run)
```

`delta_dress_fit` *Compute the Delta-k-DRESS Histogram*

Description

Compute the Δ^k-DRESS histogram by exhaustively removing all k-vertex subsets and measuring the change in edge similarity values. The result is a fixed-size histogram of edge DRESS values pooled across all $\binom{N}{k}$ subgraphs.

Usage

```
delta_dress_fit(n_vertices, sources, targets, weights = NULL, k = 0L,
                variant = DRESS_UNDIRECTED, max_iterations = 100L,
                epsilon = 1e-6, precompute = FALSE,
                keep_multisets = FALSE, offset = 0L, stride = 1L)
```

Arguments

n_vertices	Integer. Number of vertices (vertex ids must be in 0 .. n_vertices - 1).
sources	Integer vector of length E – edge source endpoints (0-based).
targets	Integer vector of length E – edge target endpoints (0-based).
weights	Numeric vector of length E – edge weights, or NULL for unweighted (all ones). Default NULL.
k	Integer. Number of vertices to remove per subset (0 = original graph, default 0).
variant	Graph variant (default DRESS_UNDIRECTED). One of DRESS_UNDIRECTED (0), DRESS_DIRECTED (1), DRESS_FORWARD (2), DRESS_BACKWARD (3).
max_iterations	Maximum number of fitting iterations per subgraph (default 100).
epsilon	Convergence threshold and histogram bin width (default 1e-6).
precompute	Logical. Pre-compute common-neighbor intercept index for faster iteration at the cost of more memory (default FALSE).
keep_multisets	Logical. If TRUE, return per-subgraph edge DRESS values in a C(N,k) x E matrix (NaN for removed edges). Default FALSE.
offset	Integer. Start index for stride-based subgraph selection (0-based). Used for distributing work across MPI ranks. Default 0L.
stride	Integer. Step size for stride-based subgraph selection (must be >= 1). Only subgraphs where index %% stride == offset are processed. Default 1L (process all subgraphs).

Value

A list with components:

histogram	Numeric vector – bin counts of edge DRESS values pooled across all subgraphs.
hist_size	Integer – number of bins (floor(dmax/epsilon) + 1; dmax = 2 unweighted).
multisets	Matrix (C(N,k) x E) of per-subgraph edge DRESS values (only present when keep_multisets = TRUE; NaN = removed edge).
num_subgraphs	Integer – C(N,k) (only present when keep_multisets = TRUE).

See Also

dress_fit for the standard DRESS computation.

Examples

```
# Triangle K3, delta-1: remove 1 vertex at a time
res <- delta_dress_fit(3L, c(0L, 1L, 2L), c(1L, 2L, 0L), k = 1L)
res$hist_size

# K4, delta-0 (original graph)
res0 <- delta_dress_fit(4L, c(0L,0L,0L,1L,1L,2L), c(1L,2L,3L,2L,3L,3L))
sum(res0$histogram)  # 6 edge values
```

DRESS *Persistent DRESS Graph Object*

Description

Create a persistent DRESS graph that stays alive across multiple `fit` / `get` (virtual-edge query) calls without rebuilding the graph each time.

Usage

```
DRESS(n_vertices, sources, targets, weights = NULL,
           variant = DRESS_UNDIRECTED,
           precompute_intercepts = FALSE)
```

Arguments

`n_vertices`	Integer. Number of vertices (vertex ids must be in `0 .. n_vertices - 1`).
`sources`	Integer vector of length E – edge source endpoints (0-based).
`targets`	Integer vector of length E – edge target endpoints (0-based).
`weights`	Optional numeric vector of length E – per-edge weights. `NULL` (default) gives every edge weight 1.
`variant`	Graph variant (default `DRESS_UNDIRECTED`). One of `DRESS_UNDIRECTED` (0), `DRESS_DIRECTED` (1), `DRESS_FORWARD` (2), `DRESS_BACKWARD` (3).
`precompute_intercepts`	Logical. Pre-compute common-neighbor index for faster iteration at the cost of more memory (default `FALSE`).

Value

An environment of class `"DRESS"` with the following methods:

`$fit(max_iterations = 100L, epsilon = 1e-6)`
: Run iterative fitting. Returns a list with `iterations` and `delta`.

`$get(u, v, max_iterations = 100L, epsilon = 1e-6, edge_weight = 1.0)`
: Query the DRESS value for an existing or virtual edge between vertices `u` and `v`.

`$result()`	Extract current results as a list with `sources`, `targets`, `edge_dress`, `edge_weight`, and `node_dress`.
`$close()`	Explicitly free the underlying C graph. Called automatically by the garbage collector if not invoked manually.

References

E. Castrillo, E. Leon, J. Gomez. Dynamic Structural Similarity on Graphs. arXiv:1805.01419, 2018.

Examples

```
g <- DRESS(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L))
g$fit(100L, 1e-6)
g$get(0L, 3L, 100L, 1e-6, 1.0)
r <- g$result()
g$close()
```

`dress_fit` *Compute DRESS Edge Similarity on Graphs*

Description

Build a DRESS graph from an edge list and run iterative fitting to compute per-edge structural similarity values.

Usage

```
dress_fit(n_vertices, sources, targets, weights = NULL,
          variant = DRESS_UNDIRECTED, max_iterations = 100L,
          epsilon = 1e-6, precompute_intercepts = FALSE)

dress_version()

DRESS_UNDIRECTED
DRESS_DIRECTED
DRESS_FORWARD
DRESS_BACKWARD
```

Arguments

`n_vertices`	Integer. Number of vertices (vertex ids must be in `0 .. n_vertices - 1`).
`sources`	Integer vector of length E – edge source endpoints (0-based).
`targets`	Integer vector of length E – edge target endpoints (0-based).
`weights`	Optional numeric vector of length E – per-edge weights. `NULL` (default) gives every edge weight 1.

variant	Graph variant (default DRESS_UNDIRECTED). One of DRESS_UNDIRECTED (0), DRESS_DIRECTED (1), DRESS_FORWARD (2), DRESS_BACKWARD (3).
max_iterations	Maximum number of fitting iterations (default 100).
epsilon	Convergence threshold – stop when the max per-edge change falls below this value (default 1e-6).
precompute_intercepts	Logical. Pre-compute common-neighbor index for faster iteration at the cost of more memory (default FALSE).

Value

A list with components:

sources	Integer vector [E] – edge source endpoints (0-based).
targets	Integer vector [E] – edge target endpoints (0-based).
edge_dress	Numeric vector [E] – DRESS similarity per edge.
edge_weight	Numeric vector [E] – variant-specific weight.
node_dress	Numeric vector [N] – per-node norm.
iterations	Integer – number of iterations performed.
delta	Numeric – final max per-edge change.

References

E. Castrillo, E. Leon, J. Gomez. Dynamic Structural Similarity on Graphs. arXiv:1805.01419, 2018.

Examples

```
# Triangle + pendant: 0-1, 1-2, 2-0, 2-3
res <- dress_fit(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L))
res$edge_dress
```

---

mpi *MPI-distributed DRESS functions.*

Description

An environment containing MPI-distributed versions of delta_dress_fit. Switch from CPU to MPI by prefixing calls with mpi$.

Details

The `mpi` environment provides:

`mpi$delta_dress_fit(...)` MPI-distributed `delta_dress_fit` (CPU backend). Same arguments plus `comm_f`.

`mpi$cuda$delta_dress_fit(...)` MPI-distributed `delta_dress_fit` (CUDA backend). Each rank runs GPU-accelerated DRESS.

MPI support requires rebuilding the package with `DRESS_MPI` (auto-detected when `mpicc` is available).

Examples

```
## Not run:
  # CPU
  r1 <- delta_dress_fit(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L), k = 1L)

  # MPI -- same signature, distributed
  r2 <- mpi$delta_dress_fit(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L), k = 1L)

  # MPI + CUDA
  r3 <- mpi$cuda$delta_dress_fit(4L, c(0L,1L,2L,2L), c(1L,2L,0L,3L), k = 1L)

## End(Not run)
```

Index