# Package ‘bigKNN’

April 1, 2026

**Type** Package

**Title** Exact Search and Graph Construction for 'bigmemory' Matrices

**Version** 0.3.0

**Date** 2026-03-26

**Author** Frederic Bertrand [aut, cre]

**Maintainer** Frederic Bertrand <frederic.bertrand@lecnam.net>

**Description** Exact nearest-neighbour and radius-search routines that operate directly on 'bigmemory::big.matrix' objects. The package streams row blocks through 'BLAS' kernels, supports self-search and external-query search, exposes prepared references for repeated queries, and can build exact k-nearest-neighbour, radius, mutual k-nearest-neighbour, and shared-nearest-neighbour graphs. Version 0.3.0 adds execution plans, serializable prepared caches, resumable streamed graph jobs, coercion helpers, exact candidate reranking, and recall summaries for evaluating approximate neighbours.

**License** GPL (>= 2)

**Depends** R (>= 3.5.0)

**Imports** Matrix, Rcpp (>= 1.0.12), methods

**Suggests** bigmemory, knitr, rmarkdown, testthat (>= 3.0.0)

**LinkingTo** Rcpp, bigmemory, BH

**Encoding** UTF-8

**NeedsCompilation** yes

**URL** https://fbertran.github.io/bigKNN/,
https://github.com/fbertran/bigKNN

**BugReports** https://github.com/fbertran/bigKNN/issues

**RoxygenNote** 7.3.3

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Repository** CRAN

**Date/Publication** 2026-04-01 08:00:26 UTC

# Contents

## as_edge_list *Coerce bigKNN outputs to edge-list form*

### Description

Coerce bigKNN outputs to edge-list form

### Usage

```
as_edge_list(x, include_distance = TRUE)
```

### Arguments

`x` A bigKNN result or graph object.

`include_distance` Logical flag controlling whether distances are kept when coercing raw kNN or radius results.

### Value

A data frame with columns `from`, `to`, and either `distance` or `weight`.

## `as_sparse_matrix` *Coerce bigKNN outputs to a sparse matrix*

### Description

Coerce bigKNN outputs to a sparse matrix

### Usage

```
as_sparse_matrix(x, include_distance = TRUE)
```

### Arguments

`x` A bigKNN result or graph object.

`include_distance` Logical flag controlling whether distances are kept when coercing raw kNN or radius results.

### Value

A `Matrix::dgCMatrix`.

## `as_triplet` *Coerce bigKNN outputs to sparse-triplet form*

### Description

Coerce bigKNN outputs to sparse-triplet form

### Usage

```
as_triplet(x, include_distance = TRUE)
```

### Arguments

`x` A bigKNN result or graph object.

`include_distance` Logical flag controlling whether distances are kept when coercing raw kNN or radius results.

### Value

A triplet list with components `i`, `j`, `x`, and `Dim`.

## count_within_radius_bigmatrix

*Count neighbours within a fixed radius*

### Description

Count neighbours within a fixed radius

### Usage

```
count_within_radius_bigmatrix(
  x,
  query = NULL,
  radius,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query)
)
```

### Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix, an external pointer referencing a big.matrix, or a prepared reference returned by knn_prepare_bigmatrix(). |
| query | Optional query source. Supply NULL for self-search, another big.matrix or external pointer for streamed queries, or a dense numeric matrix. |
| radius | Distance threshold for including a neighbour. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |
| block_size | Number of rows to process per query and reference block. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| exclude_self | Logical flag controlling whether a query row may return itself as a neighbour when query references the same matrix as x. |

### Value

An integer vector with one count per query row.

## knn_bigmatrix *Exact k-nearest neighbours for* bigmemory::big.matrix

### Description

Exact k-nearest neighbours for bigmemory::big.matrix

### Usage

```
knn_bigmatrix(
  x,
  query = NULL,
  k = 10L,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query)
)
```

### Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix, an external pointer referencing a big.matrix, or a prepared reference returned by knn_prepare_bigmatrix(). |
| query | Optional query source. Supply NULL for self-search, another big.matrix or external pointer for streamed queries, or a dense numeric matrix. |
| k | Number of neighbours to return. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |
| block_size | Number of rows to process per query and reference block. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| exclude_self | Logical flag controlling whether a query row may return itself as a neighbour when query references the same matrix as x. |

### Value

A list with components index, distance, k, metric, n_ref, n_query, exact, and backend.

## knn_graph_bigmatrix *Build an exact kNN graph from a* `bigmemory::big.matrix`

### Description

Build an exact kNN graph from a `bigmemory::big.matrix`

### Usage

```
knn_graph_bigmatrix(
  x,
  k = 10L,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  include_distance = TRUE,
  format = c("edge_list", "triplet", "dgCMatrix"),
  symmetrize = c("none", "union", "mutual"),
  exclude_self = TRUE
)
```

### Arguments

`x` A `bigmemory::big.matrix`, an external pointer referencing a `big.matrix`, or a prepared reference returned by `knn_prepare_bigmatrix()`.

`k` Number of neighbours per row.

`metric` Distance metric. Supported values are `"euclidean"`, `"sqeuclidean"`, and `"cosine"`.

`block_size` Number of rows to process per query and reference block.

`plan` Optional execution plan returned by `knn_plan_bigmatrix()`.

`include_distance`
Logical flag controlling whether kNN graph edges store distances or unit weights.

`format` Output format. One of `"edge_list"`, `"triplet"`, or `"dgCMatrix"`.

`symmetrize` How directed kNN edges should be combined. One of `"none"`, `"union"`, or `"mutual"`.

`exclude_self` Logical flag controlling whether self loops are suppressed in the directed kNN graph.

### Value

An edge list, a triplet list, or a `Matrix::dgCMatrix`, depending on the requested `format`.

## knn_graph_stream_bigmatrix *Stream a directed exact kNN graph into destination* big.matrix *objects*

### Description

Stream a directed exact kNN graph into destination `big.matrix` objects

### Usage

```
knn_graph_stream_bigmatrix(
  x,
  k,
  xpFrom,
  xpTo,
  xpValue = NULL,
  metric = "euclidean",
  plan = NULL,
  block_size = knn_default_block_size(),
  include_distance = TRUE,
  checkpoint_path = NULL
)
```

### Arguments

| | |
|---|---|
| `x` | A `bigmemory::big.matrix` or prepared reference with a stored descriptor. |
| `k` | Number of neighbours per row. |
| `xpFrom` | Writable single-column `bigmemory::big.matrix` receiving source vertex ids. |
| `xpTo` | Writable single-column `bigmemory::big.matrix` receiving target vertex ids. |
| `xpValue` | Optional writable single-column `bigmemory::big.matrix` receiving distances or unit weights. |
| `metric` | Distance metric. Supported values are `"euclidean"`, `"sqeuclidean"`, and `"cosine"`. |
| `plan` | Optional execution plan returned by `knn_plan_bigmatrix()`. |
| `block_size` | Number of rows to process per query and reference block. |
| `include_distance` | Logical flag controlling whether `xpValue` stores distances or unit weights. |
| `checkpoint_path` | Optional path for a resumable job checkpoint. |

### Value

An object of class `"bigknn_job"`.

## `knn_load_prepared` *Load a serialized prepared reference*

### Description

Load a serialized prepared reference

### Usage

```
knn_load_prepared(cache_path)
```

### Arguments

| | |
|---|---|
| `cache_path` | Path previously written by `knn_prepare_bigmatrix()` with `cache_path =`. |

### Value

An object of class `"bigknn_prepared"`.

## `knn_plan_bigmatrix` *Build an execution plan for exact search*

### Description

Build an execution plan for exact search

### Usage

```
knn_plan_bigmatrix(
  x,
  metric = "euclidean",
  memory_budget = "2GB",
  num_threads = getOption("bigKNN.num_threads", 1L),
  progress = getOption("bigKNN.progress", interactive())
)
```

### Arguments

| | |
|---|---|
| `x` | A `bigmemory::big.matrix`, external pointer, or prepared reference. |
| `metric` | Distance metric. Supported values are `"euclidean"`, `"sqeuclidean"`, and `"cosine"`. |
| `memory_budget` | Memory budget expressed in bytes or a compact size string such as `"2GB"`. |
| `num_threads` | Requested thread count forwarded to common BLAS/OpenMP environment variables during execution. |
| `progress` | Logical flag controlling progress reporting for plan-aware calls. |

## Value

An object of class "bigknn_plan".

---

# knn_prepare_bigmatrix *Prepare a* bigmemory::big.matrix *reference for repeated exact search*

---

## Description

Prepare a bigmemory::big.matrix reference for repeated exact search

## Usage

```
knn_prepare_bigmatrix(
  x,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  validate = TRUE,
  cache_path = NULL
)
```

## Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix or an external pointer referencing the reference matrix. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |
| block_size | Number of rows to process per query and reference block. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| validate | Logical flag controlling whether the preparation pass checks for finite, metric-compatible rows while building the cache. |
| cache_path | Optional path where a serializable prepared-reference cache should be written with saveRDS(). |

## Value

An object of class "bigknn_prepared" containing the reference pointer, metric-specific row cache, and metadata reused by later exact search calls.

## knn_search_prepared *Search a prepared exact reference*

### Description

Search a prepared exact reference

### Usage

```
knn_search_prepared(
  ref,
  query = NULL,
  k = 10L,
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query)
)
```

### Arguments

| | |
|---|---|
| ref | A prepared reference returned by knn_prepare_bigmatrix(). |
| query | Optional query source. Supply NULL for self-search, another big.matrix or external pointer for streamed queries, or a dense numeric matrix. |
| k | Number of neighbours to return. |
| block_size | Number of rows to process per query and reference block. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| exclude_self | Logical flag controlling whether a query row may return itself as a neighbour when query references the same matrix as the prepared reference. |

### Value

A list with components index, distance, k, metric, n_ref, n_query, exact, and backend.

## knn_search_stream_prepared *Stream prepared exact search results into destination big.matrix objects*

### Description

Stream prepared exact search results into destination big.matrix objects

## Usage

```
knn_search_stream_prepared(
  ref,
  query = NULL,
  xpIndex,
  xpDistance = NULL,
  k = 10L,
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query)
)
```

## Arguments

`ref` A prepared reference returned by `knn_prepare_bigmatrix()`.

`query` Optional query source. Supply `NULL` for self-search, another `big.matrix` or external pointer for streamed queries, or a dense numeric matrix.

`xpIndex` A writable `bigmemory::big.matrix` or external pointer that receives the 1-based neighbour indices.

`xpDistance` Optional writable `bigmemory::big.matrix` or external pointer that receives the neighbour distances.

`k` Number of neighbours to return.

`block_size` Number of rows to process per query and reference block.

`plan` Optional execution plan returned by `knn_plan_bigmatrix()`.

`exclude_self` Logical flag controlling whether a query row may return itself as a neighbour when `query` references the same matrix as the prepared reference.

## Value

A list with components `index`, `distance`, `k`, `metric`, `n_ref`, `n_query`, `exact`, and `backend`. The `index` and `distance` entries reference the supplied destination objects.

---

# `knn_stream_bigmatrix` *Stream exact k-nearest neighbours into destination* `big.matrix` *objects*

---

## Description

Stream exact k-nearest neighbours into destination `big.matrix` objects

### Usage

```
knn_stream_bigmatrix(
  x,
  query = NULL,
  xpIndex,
  xpDistance = NULL,
  k = 10L,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query)
)
```

### Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix, an external pointer referencing a big.matrix, or a prepared reference returned by knn_prepare_bigmatrix(). |
| query | Optional query source. Supply NULL for self-search, another big.matrix or external pointer for streamed queries, or a dense numeric matrix. |
| xpIndex | A writable bigmemory::big.matrix or external pointer that receives the 1-based neighbour indices. |
| xpDistance | Optional writable bigmemory::big.matrix or external pointer that receives the neighbour distances. |
| k | Number of neighbours to return. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |
| block_size | Number of rows to process per query and reference block. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| exclude_self | Logical flag controlling whether a query row may return itself as a neighbour when query references the same matrix as x. |

### Value

A list with components index, distance, k, metric, n_ref, n_query, exact, and backend. The index and distance entries reference the supplied destination objects.

---

## knn_validate_prepared *Validate a prepared reference*

---

### Description

Validate a prepared reference

### Usage

```
knn_validate_prepared(ref)
```

### Arguments

ref A prepared reference returned by knn_prepare_bigmatrix() or knn_load_prepared().

### Value

Invisibly returns TRUE when the prepared reference is valid.

---

## mutual_knn_graph_bigmatrix

*Build an exact mutual kNN graph from a* bigmemory::big.matrix

---

### Description

Build an exact mutual kNN graph from a bigmemory::big.matrix

### Usage

```
mutual_knn_graph_bigmatrix(
  x,
  k = 10L,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  include_distance = TRUE,
  format = c("edge_list", "triplet", "dgCMatrix")
)
```

### Arguments

x A bigmemory::big.matrix, an external pointer referencing a big.matrix, or a prepared reference returned by knn_prepare_bigmatrix().

k Number of neighbours per row.

metric Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine".

block_size Number of rows to process per query and reference block.

plan Optional execution plan returned by knn_plan_bigmatrix().

include_distance
Logical flag controlling whether graph edges store distances or unit weights.

format Output format. One of "edge_list", "triplet", or "dgCMatrix".

### Value

An edge list, a triplet list, or a Matrix::dgCMatrix, depending on the requested format.

## radius_bigmatrix — *Exact radius search for* bigmemory::big.matrix

### Description

Exact radius search for bigmemory::big.matrix

### Usage

```
radius_bigmatrix(
  x,
  query = NULL,
  radius,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query),
  sort = TRUE
)
```

### Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix, an external pointer referencing a big.matrix, or a prepared reference returned by knn_prepare_bigmatrix(). |
| query | Optional query source. Supply NULL for self-search, another big.matrix or external pointer for streamed queries, or a dense numeric matrix. |
| radius | Distance threshold for including a neighbour. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |
| block_size | Number of rows to process per query and reference block. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| exclude_self | Logical flag controlling whether a query row may return itself as a neighbour when query references the same matrix as x. |
| sort | Logical flag controlling whether each query's matches are sorted by distance and then by index. |

### Value

A list with components index, distance, offset, n_match, radius, metric, n_ref, n_query, exact, and backend.

## radius_graph_bigmatrix *Build an exact radius graph from a* `bigmemory::big.matrix`

### Description

Build an exact radius graph from a `bigmemory::big.matrix`

### Usage

```
radius_graph_bigmatrix(
  x,
  radius,
  metric = "euclidean",
  plan = NULL,
  block_size = knn_default_block_size(),
  include_distance = TRUE,
  format = c("edge_list", "triplet", "dgCMatrix"),
  symmetrize = c("none", "union", "mutual"),
  exclude_self = TRUE,
  sort = TRUE
)
```

### Arguments

| | |
|---|---|
| `x` | A `bigmemory::big.matrix`, external pointer, or prepared reference. |
| `radius` | Distance threshold for including an edge. |
| `metric` | Distance metric. Supported values are `"euclidean"`, `"sqeuclidean"`, and `"cosine"`. |
| `plan` | Optional execution plan returned by `knn_plan_bigmatrix()`. |
| `block_size` | Number of rows to process per query and reference block. |
| `include_distance` | Logical flag controlling whether graph edges store distances or unit weights. |
| `format` | Output format. One of `"edge_list"`, `"triplet"`, or `"dgCMatrix"`. |
| `symmetrize` | How directed radius edges should be combined. One of `"none"`, `"union"`, or `"mutual"`. |
| `exclude_self` | Logical flag controlling whether self loops are suppressed. |
| `sort` | Logical flag controlling whether each query's matches are sorted by distance and then by index. |

### Value

A graph representation in the requested `format`.

## radius_stream_bigmatrix

*Stream exact radius-search results into destination* big.matrix *objects*

### Description

Stream exact radius-search results into destination big.matrix objects

### Usage

```
radius_stream_bigmatrix(
  x,
  query = NULL,
  xpIndex,
  xpDistance = NULL,
  xpOffset,
  radius,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  exclude_self = is.null(query),
  sort = TRUE
)
```

### Arguments

`x` A bigmemory::big.matrix, an external pointer referencing a big.matrix, or a prepared reference returned by knn_prepare_bigmatrix().

`query` Optional query source. Supply NULL for self-search, another big.matrix or external pointer for streamed queries, or a dense numeric matrix.

`xpIndex` A writable single-column bigmemory::big.matrix or external pointer that receives flattened 1-based neighbour indices.

`xpDistance` Optional writable single-column bigmemory::big.matrix or external pointer that receives flattened neighbour distances.

`xpOffset` A writable single-column bigmemory::big.matrix or external pointer that receives 1-based offsets into the flattened match vectors.

`radius` Distance threshold for including a neighbour.

`metric` Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine".

`block_size` Number of rows to process per query and reference block.

`plan` Optional execution plan returned by knn_plan_bigmatrix().

`exclude_self` Logical flag controlling whether a query row may return itself as a neighbour when query references the same matrix as x.

`sort` Logical flag controlling whether each query's matches are sorted by distance and then by index.

## Value

A list with components index, distance, offset, n_match, radius, metric, n_ref, n_query, exact, and backend. The index, distance, and offset entries reference the supplied destination objects.

---

radius_stream_job_bigmatrix

*Stream exact radius-search results into destination* big.matrix *objects with checkpoints*

---

## Description

Stream exact radius-search results into destination big.matrix objects with checkpoints

## Usage

```
radius_stream_job_bigmatrix(
  x,
  query = NULL,
  xpIndex,
  xpDistance = NULL,
  xpOffset,
  radius,
  metric = "euclidean",
  plan = NULL,
  block_size = knn_default_block_size(),
  exclude_self = is.null(query),
  sort = TRUE,
  checkpoint_path = NULL
)
```

## Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix or prepared reference with a stored descriptor. |
| query | Optional query source. Supply NULL for self-search, another big.matrix, or a dense/sparse matrix. |
| xpIndex | A writable single-column bigmemory::big.matrix that receives flattened 1-based neighbour indices. |
| xpDistance | Optional writable single-column bigmemory::big.matrix that receives flattened neighbour distances. |
| xpOffset | A writable single-column bigmemory::big.matrix that receives 1-based offsets into the flattened output vectors. |
| radius | Distance threshold for including a neighbour. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |

| | |
|---|---|
| `plan` | Optional execution plan returned by `knn_plan_bigmatrix()`. |
| `block_size` | Number of query rows to process per block. |
| `exclude_self` | Logical flag controlling whether self matches are removed when `query` references the same source as `x`. |
| `sort` | Logical flag controlling whether each query's matches are sorted by distance and then by index. |
| `checkpoint_path` | Optional path for a resumable job checkpoint. |

## Value

An object of class `"bigknn_job"`.

# `recall_against_exact` *Compare approximate neighbours to exact truth*

## Description

Compare approximate neighbours to exact truth

## Usage

```
recall_against_exact(exact, approx_index, k = NULL)
```

## Arguments

| | |
|---|---|
| `exact` | Exact kNN output or index matrix. |
| `approx_index` | Approximate neighbour index matrix or result object. |
| `k` | Optional number of neighbours to compare. |

## Value

An object of class `"bigknn_recall"`.

rerank_candidates_bigmatrix *Rerank candidate neighbours exactly against a* bigmemory::big.matrix

## Description

Rerank candidate neighbours exactly against a bigmemory::big.matrix

## Usage

```
rerank_candidates_bigmatrix(
  x,
  query,
  candidate_index,
  metric = "euclidean",
  top_k = NULL,
  plan = NULL,
  block_size = knn_default_block_size(),
  exclude_self = is.null(query)
)
```

## Arguments

| | |
|---|---|
| x | A bigmemory::big.matrix or prepared reference with a stored descriptor. |
| query | Query source. Supply NULL to rerank self-query candidates. |
| candidate_index | Candidate neighbour indices supplied as a matrix, bigmemory::big.matrix, or kNN result object. |
| metric | Distance metric. Supported values are "euclidean", "sqeuclidean", and "cosine". |
| top_k | Number of reranked neighbours to return. Defaults to all supplied candidate columns. |
| plan | Optional execution plan returned by knn_plan_bigmatrix(). |
| block_size | Number of query rows to process at a time. |
| exclude_self | Logical flag controlling whether self ids are removed when query = NULL. |

## Value

An object of class "bigknn_knn_result".

## resume_knn_job *Resume a checkpointed bigKNN job*

### Description

Resume a checkpointed bigKNN job

### Usage

```
resume_knn_job(checkpoint_path)
```

### Arguments

checkpoint_path
: Path previously created by `knn_graph_stream_bigmatrix()` or `radius_stream_job_bigmatrix()`.

### Value

An object of class "`bigknn_job`".

## snn_graph_bigmatrix *Build an exact shared-nearest-neighbour graph from a* `bigmemory::big.matrix`

### Description

Build an exact shared-nearest-neighbour graph from a `bigmemory::big.matrix`

### Usage

```
snn_graph_bigmatrix(
  x,
  k = 10L,
  metric = "euclidean",
  block_size = knn_default_block_size(),
  plan = NULL,
  weight = c("count", "jaccard"),
  format = c("edge_list", "triplet", "dgCMatrix")
)
```

## Arguments

`x` A `bigmemory::big.matrix`, an external pointer referencing a `big.matrix`, or a prepared reference returned by `knn_prepare_bigmatrix()`.

`k` Number of neighbours per row in the underlying exact kNN search.

`metric` Distance metric. Supported values are `"euclidean"`, `"sqeuclidean"`, and `"cosine"`.

`block_size` Number of rows to process per query and reference block.

`plan` Optional execution plan returned by `knn_plan_bigmatrix()`.

`weight` Shared-nearest-neighbour weight definition. One of `"count"` or `"jaccard"`.

`format` Output format. One of `"edge_list"`, `"triplet"`, or `"dgCMatrix"`.

## Value

An edge list, a triplet list, or a `Matrix::dgCMatrix`, depending on the requested `format`.

# Index