Package: cellNexus (via r-universe)

November 25, 2026

Type Package

Title Queries the Human Cell Atlas

Version 1.0.0

Description Provides access to a copy of the Human Cell Atlas, but with harmonised metadata. This allows for uniform querying across numerous datasets within the Atlas using common fields such as cell type, tissue type, and patient ethnicity. Usage involves first querying the metadata table for cells of interest, and then downloading the corresponding cells into one of several supported formats, including SingleCellExperiment, Seurat, or pseudobulk SummarizedExperiment objects. Note: The Human Cell Atlas data accessed through this package is subject to its own licensing terms (typically Creative Commons Attribution or similar open data licenses as specified by the Human Cell Atlas Data Use Agreement), which differ from the package license (GPL-2). See the package documentation for details.

License GPL-2

Depends R (>= 4.5.0)

Imports dplyr, SummarizedExperiment, SingleCellExperiment, purrr (>= 1.0.0), BiocGenerics, glue, HDF5Array, DBI, tools, httr, curl, cli, lifecycle, methods, rlang, S4Vectors, tibble, utils, dbplyr (>= 2.3.0), duckdb, checkmate, shiny, shinyWidgets, zellkonverter, anndataR, stringr, Matrix, rclipboard

Suggests BiocStyle, knitr, rmarkdown, testthat, basilisk, arrow, reticulate, cellxgenedp, spelling, forcats, ggplot2, rprojroot, openssl, DelayedArray, tidybulk, BiocParallel, SeuratObject, Seurat, preprocessCore

Biarch true

biocViews AssayDomain, Infrastructure, SingleCell, RNASeq, DifferentialExpression, GeneExpression, Normalization, Clustering, QualityControl, Sequencing, Transcription, Transcriptomics, ShinyApps

Encoding UTF-8

URL https://github.com/MangiolaLaboratory/cellNexus,
https://mangiolalaboratory.github.io/cellNexus/

BugReports https://github.com/MangiolaLaboratory/cellNexus/issues

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Collate 'utils.R' 'counts.R' 'counts_per_million.R' 'data.R' 'dev.R'
'interface_app.R' 'metadata.R' 'seurat.R' 'unharmonised.R'
'zzz.R'

Language en-US

SystemRequirements Python

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

Config/pak/sysreqs cmake make libicu-dev libpng-dev libuv1-dev libssl-dev python3 zlib1g-dev

Repository Bioconductor 3.24

Date/Publication 2026-11-23 22:56:15 UTC

RemoteUrl https://github.com/bioc/cellNexus

RemoteRef RELEASE_3_24

RemoteSha 1dc1f58715eb4b8eb723622c9c0369ca600d913e

Contents

cellNexus-package . . . 3
create_interface_app . . . 4
get_atlas_versions . . . 5
get_cell_communication_strength . . . 6
get_counts_per_million . . . 7
get_default_cache_dir . . . 8
get_metadata . . . 9
get_metadata_url . . . 11
get_pseudobulk . . . 12
get_seurat . . . 14
get_single_cell_experiment . . . 15
get_SingleCellExperiment . . . 17
keep_quality_cells . . . 18
organize_inputs . . . 19
pbmc3k_sce . . . 20
SAMPLE_DATABASE_URL . . . 21
ui_choices . . . 22

Index **24**

cellNexus-package *cellNexus: Query Interface for the Human Cell Atlas*

Description

cellNexus provides a query interface for programmatic exploration and retrieval of the harmonised, curated and reannotated CELLxGENE single-cell Human Cell Atlas. The package allows users to query metadata and download single-cell RNA sequencing data in various formats including SingleCellExperiment, Seurat, and pseudobulk SummarizedExperiment objects.

Value

The cellNexus package (invisibly).

Getting Started

To get started with cellNexus, first load the package and retrieve the metadata:

```
library(cellNexus)
metadata <- get_metadata()
```

Then filter the metadata to find cells of interest and download the data:

```
filtered_metadata <- metadata |>
    dplyr::filter(
        imputed_ethnicity == "African" &
        cell_type_unified_ensemble %LIKE% "%cd4%"
    )

sce <- get_single_cell_experiment(filtered_metadata)
```

Main Functions

get_metadata Retrieve and query the harmonised metadata

get_single_cell_experiment Download data as SingleCellExperiment objects

get_seurat Download data as Seurat objects

get_pseudobulk Download aggregated pseudobulk data

Data Licensing

Important: The Human Cell Atlas data accessed through this package is subject to its own licensing terms, which differ from the package license. The cellNexus package itself is licensed under GPL-2. However, the underlying Human Cell Atlas data is typically licensed under Creative Commons Attribution (CC-BY) or similar open data licenses as specified by the Human Cell Atlas Data Use Agreement. Users should review the specific licensing terms for any datasets they access through this package. For more information, see the Human Cell Atlas Data Portal: https://data.humancellatlas.org/about

Vignettes

See `vignette("cellNexus", package = "cellNexus")` for a comprehensive introduction to using the package.

Source

Mangiola et al.,2023

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

`create_interface_app` *Create a Shiny app that allows users to generate filtering & retreival code for cellNexus*

Description

Create a Shiny app that allows users to generate filtering & retreival code for cellNexus

Usage

```
create_interface_app(ui_choices, return_as_list = FALSE)
```

Arguments

`ui_choices`	A list of pre-computed unique values for each filterable column in the metadata.
`return_as_list`	If TRUE, returns a list with 'ui' and 'server' components instead of a Shiny app object.

Value

A Shiny app that allows users to filter cellNexus metadata and generate code for retrieval in the selected format.

Author(s)

Jared Andrews

Source

Mangiola et al.,2023

Examples

```
get_default_cache_dir()

# Create the interface app with metadata
metadata <- get_metadata(cloud_metadata = SAMPLE_DATABASE_URL)
app <- create_interface_app(metadata)
# Run the app
shiny::runApp(app)
```

---

get_atlas_versions *Returns the atlas version changelog as a tibble*

Description

Downloads the `atlas_versions.parquet` registry from the cellNexus metadata store, caches it locally, and returns it as an in-memory tibble. Each row describes one atlas data release and its relationship to a CellxGene Census snapshot.

Usage

```
get_atlas_versions(cache = tempdir())
```

Arguments

cache	Optional character scalar. A local directory used to cache the downloaded parquet file. Defaults to a temporary directory to separate from the main cache directory.

Details

The `atlas_id` column in this table corresponds directly to the `atlas_id` column returned by `get_metadata()`, so you can join them to find which Census snapshot any cell in your query came from.

Value

A tibble with columns:

atlas_id Atlas version identifier, e.g. `"cellxgene_2024/0.1.0"`. Matches the `atlas_id` column in `get_metadata()`.

census_version The CellxGene Census snapshot this atlas was built from, e.g. `"01-07-2024"`.

change_type One of `"initial"`, `"patch"`, `"minor"`, or `"major"`. See `ATLAS_VERSIONS.md` for the conventions standards.

description Summary text of what changed in this release.

modified_at Modification date as a character scalar (`"YYYY-MM-DD"`). By default use `Sys.Date()`

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

See Also

- CellxGene Census data releases (LTS): https://chanzuckerberg.github.io/cellxgene-census/cellxgene_census_docsite_data_release_info.html

Examples

```
get_atlas_versions()
```

`get_cell_communication_strength` *Retrieve cellNexus cell communication ligand–receptor strength as a data frame.*

Description

Downloads a parquet database of the cell communication strength to a local cache, and then opens it as a data frame. It can then be filtered.

Usage

```
get_cell_communication_strength(
  cloud_metadata = get_metadata_url("cellNexus_lr_signaling_pathway_strength.parquet"),
  local_metadata = NULL,
  cache_directory = get_default_cache_dir(),
  use_cache = TRUE
)
```

Arguments

`cloud_metadata`	Character vector of any length. HTTP URL/URLs pointing to the name and location of parquet database/databases. By default, it points to cell communication metadata in cellNexus ARDC Nectar Research Cloud. Assign `NULL` to query local_metadata only if exists.
`local_metadata`	Optional character vector of any length representing the local path of parquet database(s).

cache_directory
: Optional character vector of length 1. A file path on your local system to a directory (not a file) that will be used to store `metadata.parquet`

use_cache
: Optional logical scalar. If `TRUE` (the default), and this function has been called before with the same parameters, then a cached reference to the table will be returned. If `FALSE`, a new connect138/4.7ion will be created no matter what.

Details

The returned table integrates three levels of cell communication inference from `CellChat`, for each sample: (i) ligand–receptor–level communication (`lr_prob`, `lr_pval`), (ii) pathway-level aggregated signaling (`pathway_prob`, `pathway_pval`), (iii) cell-pair–level summaries of communication breadth (`interaction_count` - number of significant LR interactions) and intensity (`interaction_weight` - overall communication strength).

Together, these metrics allow simultaneous assessment of signaling specificity, pathway dominance, and global communication structure between cell populations.

Value

A lazy data.frame subclass containing the metadata. You can interact with this object using most standard dplyr functions. For string matching, it is recommended that you use `stringr::str_like` to filter character columns, as `stringr::str_match` will not work.

Examples

```
# For fast build purpose only, you do not need to specify anything in the function.
communication_meta <- get_cell_communication_strength(
  cloud_metadata = get_metadata_url(
    "cellNexus_lr_signaling_pathway_strength_DEMO.parquet"
  )
)
```

---

get_counts_per_million

Generating counts per million from a SingleCellExperiment object

---

Description

Generating counts per million from a SingleCellExperiment object

Usage

```
get_counts_per_million(sce, output_file)
```

Arguments

sce
: A SingleCellExperiment object

output_file
: A character vector of CPM Anndata file path

Value

A directory stores counts per million Anndata

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
data(pbmc3k_sce)
get_counts_per_million(pbmc3k_sce, tempfile(fileext = ".h5ad"))
```

---

get_default_cache_dir *Returns the default cache directory with a version number*

Description

Returns the default cache directory with a version number

Usage

```
get_default_cache_dir()
```

Value

A length one character vector.

Cache integrity checking

By default, every file in the cache is verified against its remote ETag (MD5) before being reused. This catches partial downloads left by interrupted sessions. You should only disable the check when the cache is known to be intact.

```
options(cellNexus.check_cache_integrity = FALSE)
```

To make this permanent, add the line to your '`~/.Rprofile`'.

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
get_metadata(cloud_metadata = SAMPLE_DATABASE_URL, cache_directory = get_default_cache_dir())
```

---

get_metadata *Gets the CellNexus metadata as a data frame.*

Description

Downloads a parquet database of the Human Cell Atlas metadata to a local cache, and then opens it as a data frame. It can then be filtered and passed into `get_single_cell_experiment()` to obtain a `SingleCellExperiment::SingleCellExperiment`

Usage

```
get_metadata(
  cloud_metadata = get_metadata_url("hca_2024"),
  local_metadata = NULL,
  cache_directory = get_default_cache_dir(),
  use_cache = TRUE
)
```

Arguments

`cloud_metadata`	Optional character vector of any length. HTTP URL/URLs pointing to the name and location of parquet database/databases. By default, it points to cellNexus ARDC Nectar Research Cloud. Assign `NULL` to query local_metadata only if exists.
`local_metadata`	Optional character vector of any length representing the local path of parquet database(s).
`cache_directory`	Optional character vector of length 1. A file path on your local system to a directory (not a file) that will be used to store metadata.
`use_cache`	Optional logical scalar. If `TRUE` (the default), and this function has been called before with the same parameters, then a cached reference to the table will be returned. If `FALSE`, a new connection will be created no matter what.

Details

The metadata was collected from the Bioconductor package `cellxgenedp`. `vignette("using_cellxgenedp", package="cellxgenedp")` provides an overview of the columns in the metadata. The data for which the column `organism_name` included "Homo sapiens" was collected collected from `cellxgenedp`.

The columns `dataset_id` and `file_id_cellNexus_single_cell` link the datasets explorable through `cellNexus` and `cellxgenedp`to the CELLxGENE portal.

Our representation, harmonises the metadata at dataset, sample and cell levels, in a unique coherent database table.

Field definitions for the CELLxGENE schema follow the CELLxGENE schema 5.1.0.

Through harmonisation and curation we introduced custom columns not present in the original CELLxGENE metadata:

`sample_id`: Sample identifier. `age_days`: Donor age in days. `empty_droplet`: Whether a cell is called an empty droplet from expressed-gene count per sample (default threshold 200; targeted panels may differ). `alive`: Whether a cell passes viability / mitochondrial QC. `scDblFinder.class`: Doublet, singlet, or unknown (`scDblFinder` default parameters). `cell_type_unified_ensemble`: Consensus immune identity from Azimuth and SingleR (Blueprint, Monaco). `cell_annotation_azimuth_l2`: Azimuth cell annotation. `cell_annotation_blueprint_singler`: SingleR annotation (Blueprint). `cell_annotation_monaco_singler`: SingleR annotation (Monaco). `subsets_Mito_percent`: Percent of each cell's total counts coming from mitochondrial genes in a sample. `subsets_Ribo_percent`: Percent of each cell's total counts coming from ribosomal genes in a sample. `high_mitochondrion`: TRUE if the cell's mitochondrial percent exceeds the QC cutoff. `high_ribosome`: TRUE if the cell's ribosomal percent exceeds the QC cutoff. `count_upper_bound`: Count capping threshold used in counts transformation. `inversed_inferred_distribution`: Transformation method used in pre-processing pipeline. `nfeature_expressed_thresh`: Threshold of the number of expressed features per cell. `is_immune`: Curated logical flag for immune-cell context. `file_id_cellNexus_single_cell`: Internal file id for single-cell layers. `file_id_cellNexus_pseudobulk`: Internal file id for pseudobulk layers. `sample_id`: Harmonised sample identifier. `nCount_RNA`: Total RNA counts per cell (sample-aware). `nFeature_expressed_in_sample`: Number of expressed features per cell. `ethnicity_flagging_score`: Supporting score for ethnicity imputation. `low_confidence_ethnicity`: Supporting flag for low-confidence ethnicity calls. `.aggregated_cells`: Post-QC cells combined into each pseudobulk sample. `imputed_ethnicity`: Imputed ethnicity label. `atlas_id`: cellNexus atlas release identifier (internal use).

For all fields definitions, please refer to our documentation site

Possible cache path issues

If your default R cache path includes non-standard characters (e.g. dash because of your user or organisation name), the following error can occur.

```
Error in `db_query_fields.DBIConnection()`: ! Can't query fields. Caused by
error: ! Parser Error: syntax error at or near "/" LINE 2: FROM
/Users/bob/Library/Caches...
```

The solution is to choose a different cache, for example

```
get_metadata(cache_directory = path.expand('~'))
```

Value

A lazy data.frame subclass containing the metadata. You can interact with this object using most standard dplyr functions. For string matching, it is recommended that you use `stringr::str_like` to filter character columns, as `stringr::str_match` will not work.

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
library(dplyr)
# For fast build purpose only, you do not need to specify anything in cloud_metadata.
filtered_metadata <- get_metadata(cloud_metadata = SAMPLE_DATABASE_URL) |>
  filter(
    imputed_ethnicity == "African" &
      cell_type_unified_ensemble %LIKE% "%cd14%"
  )
```

`get_metadata_url` *Returns the URLs for all metadata files*

Description

Returns the URLs for all metadata files

Usage

```
get_metadata_url(databases = c("hca_2024"))
```

Arguments

`databases`	A character vector of atlas aliases or raw parquet filenames. Recognised aliases are `"hca_2024"` Raw filenames (e.g. `"atlas_versions.parquet"`) are passed through unchanged and are intended for internal package use.

Value

A character vector of URLs to parquet files

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
get_metadata_url("hca_2024")
```

`get_pseudobulk` *Gets a Pseudobulk from curated metadata*

Description

Given a data frame of Curated Atlas metadata obtained from `get_metadata()`, returns a `SummarizedExperiment::Summari` object corresponding to the samples in that data frame

Usage

```
get_pseudobulk(
  data,
  assays = "counts",
  cell_aggregation = "pseudobulk",
  cache_directory = get_default_cache_dir(),
  repository = COUNTS_URL,
  features = NULL,
  as_SummarizedExperiment = FALSE,
  download_only = FALSE
)
```

Arguments

`data`	A data frame containing, at minimum, `cell_id`, `file_id_cellNexus_pseudobulk`, `sample_id`, `cell_type_unified_ensemble`, `atlas_id` columns, which correspond to a single cell ID, file subdivision for internal use, a singlel cell sample ID, harmonised cell type, and atlas name in format (e.g cellxgene_2024/0.1.0) for internal use. They can be obtained from the `get_metadata()` function. Use `get_atlas_versions()` to download atlas versions data frame.
`assays`	A character vector specifying the desired assay(s) to be requested. The default setting retrieves only the counts assay.

`cell_aggregation`
: A character vector that specifies which cell aggregation strategy should be applied. This will create a corresponding subdirectory in the cache directory.

`cache_directory`
: An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied.

`repository`
: A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored.

`features`
: An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned. When provided, the returned object will contain exactly the requested features (row order preserved), and any experiments/samples that do not contain all requested features are dropped. This preserves the full set of requested features at the cost of potentially fewer samples. A warning is emitted when samples are dropped.

`as_SummarizedExperiment`
: If `TRUE`, coerce the result to a `SummarizedExperiment`. Note that `as(x, "SummarizedExperiment")` drops feature rownames; `get_pseudobulk()` restores them after coercion. Ignored when `download_only = TRUE`.

`download_only`
: Logical scalar. When `TRUE`, remote files are synchronised to `cache_directory` but the data is not read or assembled into an in-memory object. Returns `invisible(NULL)`. Useful for pre-fetching datasets only.

Details

Columns in `data` that are constant within each `sample_id` x `cell_type_unified_ensemble` combination (including user-added annotations) are retained in `colData`. Cell-level columns are dropped via internal `keep_specific_annotation_columns()`.

Value

By default, a `SingleCellExperiment` object. If `as_SummarizedExperiment` is `TRUE`, a `SummarizedExperiment` object. Returns `invisible(NULL)` when `download_only = TRUE`.

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
# Use the lightweight sample database URL (for fast checks during development only)
meta <- get_metadata(cloud_metadata = cellNexus::SAMPLE_DATABASE_URL) |>
  keep_quality_cells() |>
```

```
  dplyr::filter(cell_type_unified_ensemble == "epithelial")
pseudobulk <- meta |> get_pseudobulk()
```

`get_seurat` *Given a data frame of HCA metadata, returns a Seurat object corresponding to the samples in that data frame*

Description

Given a data frame of HCA metadata, returns a Seurat object corresponding to the samples in that data frame

Usage

```
get_seurat(..., download_only = FALSE)
```

Arguments

`...`	Arguments passed on to `get_single_cell_experiment`
	`data` A data frame containing, at minimum, `cell_id`, `file_id_cellNexus_single_cell` and `atlas_id` columns, which correspond to a single cell ID, file subdivision for internal use, and atlas name in format (e.g cellxgene_2024/0.1.0) for internal use. They can be obtained from the `get_metadata()` function. Use `get_atlas_versions()` to download atlas versions data frame.
	`assays` A character vector specifying the desired assay(s) to be requested. Valid elements include "counts", "cpm", "rank", and "sct" for single-cell analyses The default setting retrieves only the counts assay. If your analysis involves a smaller set of genes, consider using the "cpm" assay. The "rank" assay is suited for signature calculations across millions of cells.
	`cell_aggregation` A character vector that specifies which cell aggregation strategy should be applied. This will create a corresponding subdirectory in the cache directory. Single cell level is applied by default.
	`cache_directory` An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied.
	`repository` A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored.
	`features` An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned.
`download_only`	Logical scalar. When `TRUE`, remote files are synchronised to `cache_directory` but not read or assembled. Returns `invisible(NULL)`. See `get_single_cell_experiment()` for details.

Value

A Seurat object containing the same data as a call to `get_single_cell_experiment()`, or `invisible(NULL)` when `download_only = TRUE`. All requested assays are present in the returned object under their original names (e.g. `"counts"`, `"cpm"`).

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
# Use the lightweight sample database URL (for fast checks during development only)
meta <- get_metadata(cloud_metadata = cellNexus::SAMPLE_DATABASE_URL) |> head(2)
seurat <- get_seurat(meta)
```

---

get_single_cell_experiment

Gets a SingleCellExperiment from curated metadata

Description

Given a data frame of Curated Atlas metadata obtained from `get_metadata()`, returns a `SingleCellExperiment::SingleCe` object corresponding to the samples in that data frame

Usage

```
get_single_cell_experiment(
  data,
  assays = "counts",
  cell_aggregation = "",
  cache_directory = get_default_cache_dir(),
  repository = COUNTS_URL,
  features = NULL,
  download_only = FALSE
)
```

Arguments

`data`	A data frame containing, at minimum, `cell_id`, `file_id_cellNexus_single_cell` and `atlas_id` columns, which correspond to a single cell ID, file subdivision for internal use, and atlas name in format (e.g cellxgene_2024/0.1.0) for internal use. They can be obtained from the `get_metadata()` function. Use `get_atlas_versions()` to download atlas versions data frame.

assays — A character vector specifying the desired assay(s) to be requested. Valid elements include "counts", "cpm", "rank", and "sct" for single-cell analyses The default setting retrieves only the counts assay. If your analysis involves a smaller set of genes, consider using the "cpm" assay. The "rank" assay is suited for signature calculations across millions of cells.

cell_aggregation — A character vector that specifies which cell aggregation strategy should be applied. This will create a corresponding subdirectory in the cache directory. Single cell level is applied by default.

cache_directory — An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied.

repository — A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored.

features — An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned.

download_only — Logical scalar. When TRUE, remote files are synchronised to cache_directory but the data is not read or assembled into an in-memory object. Returns invisible(NULL). Useful for pre-fetching datasets only.

Value

A SingleCellExperiment object, or invisible(NULL) when download_only = TRUE.

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
# Use the lightweight sample database URL (for fast checks during development only)
meta <- get_metadata(cloud_metadata = cellNexus::SAMPLE_DATABASE_URL) |> head(2)
sce <- get_single_cell_experiment(meta)
```

`get_SingleCellExperiment` *Gets a SingleCellExperiment from curated metadata*

Description

Given a data frame of Curated Atlas metadata obtained from `get_metadata()`, returns a `SingleCellExperiment::SingleC` object corresponding to the samples in that data frame

Usage

```
get_SingleCellExperiment(...)
```

Arguments

`...`	Arguments passed on to `get_single_cell_experiment`
	`data` A data frame containing, at minimum, `cell_id`, `file_id_cellNexus_single_cell` and `atlas_id` columns, which correspond to a single cell ID, file subdivision for internal use, and atlas name in format (e.g cellxgene_2024/0.1.0) for internal use. They can be obtained from the `get_metadata()` function. Use `get_atlas_versions()` to download atlas versions data frame.
	`assays` A character vector specifying the desired assay(s) to be requested. Valid elements include "counts", "cpm", "rank", and "sct" for single-cell analyses The default setting retrieves only the counts assay. If your analysis involves a smaller set of genes, consider using the "cpm" assay. The "rank" assay is suited for signature calculations across millions of cells.
	`cell_aggregation` A character vector that specifies which cell aggregation strategy should be applied. This will create a corresponding subdirectory in the cache directory. Single cell level is applied by default.
	`cache_directory` An optional character vector of length one. If provided, it should indicate a local file path where any remotely accessed files should be copied.
	`repository` A character vector of length one. If provided, it should be an HTTP URL pointing to the location where the single cell data is stored.
	`features` An optional character vector of features (ie genes) to return the counts for. By default counts for all features will be returned.
	`download_only` Logical scalar. When `TRUE`, remote files are synchronised to `cache_directory` but the data is not read or assembled into an in-memory object. Returns `invisible(NULL)`. Useful for pre-fetching datasets only.

Value

A `SingleCellExperiment` object.

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
# Use the lightweight sample database URL (for fast checks during development only)
meta <- get_metadata(cloud_metadata = cellNexus::SAMPLE_DATABASE_URL) |> head(2)
sce <- get_single_cell_experiment(meta)
```

---

keep_quality_cells *Keep high-quality cells based on QC columns*

Description

Keep high-quality cells based on QC columns

Usage

```
keep_quality_cells(
  data,
  empty_droplet_col = "empty_droplet",
  alive_col = "alive",
  doublet_col = "scDblFinder.class",
  nfeature_col = "feature_count",
  min_features = 5000L
)
```

Arguments

data	A data frame or tibble containing single-cell metadata.
empty_droplet_col	A string specifying the column name that indicates empty droplets (default: "empty_droplet"). Expected logical vector
alive_col	A string specifying the column name that indicates whether cells are alive (default: "alive"). Expected logical vector
doublet_col	A string specifying the column name that indicates doublets (default: "scDblFinder.class"). Expected character vector: "doublet" and/or "singlet" and/or "unknown".
nfeature_col	Column containing the number of detected features per dataset
min_features	Minimum number of detected features required.

Value

A filtered data frame containing only cells that pass all QC checks.

Source

Mangiola et al.,2023

Examples

```
get_metadata(cloud_metadata = SAMPLE_DATABASE_URL, cache_directory = tempdir()) |>
  head(2) |>
  keep_quality_cells()
```

`organize_inputs` *Organize arbitrary Shiny inputs into a grid layout*

Description

Organize arbitrary Shiny inputs into a grid layout

Usage

```
organize_inputs(
  tag.list,
  id = NULL,
  title = NULL,
  tack = NULL,
  columns = NULL,
  rows = NULL
)
```

Arguments

`tag.list`	A tagList containing UI inputs or a named list containing multiple tagLists containing UI inputs.
`id`	An optional ID for the tabsetPanel if a named list is provided.
`title`	An optional title for the grid, should be a UI element, e.g. h3("Title").
`tack`	An optional UI input to tack onto the end of the grid.
`columns`	Number of columns.
`rows`	Number of rows.

Value

A Shiny tagList with inputs organized into a grid, optionally nested inside a tabsetPanel.

Author(s)

Jared Andrews

Source

Mangiola et al.,2023

Examples

```
library(shiny)
# Example 1: Basic usage with a simple grid
ui.inputs <- tagList(
  textInput("name", "Name"),
  numericInput("age", "Age", value = 30),
  selectInput("gender", "Gender", choices = c("Male", "Female", "Other"))
)
organize_inputs(ui.inputs, columns = 2, rows = 2)

# Example 2: Using a named list to create tabs
ui.inputs.tabs <- list(
  Personal = tagList(
    textInput("firstname", "First Name"),
    textInput("lastname", "Last Name")
  ),
  Settings = tagList(
    checkboxInput("newsletter", "Subscribe to newsletter", value = TRUE),
    sliderInput("volume", "Volume", min = 0, max = 100, value = 50)
  )
)
organize_inputs(ui.inputs.tabs, columns = 2)

# Example 3: Adding an additional UI element with 'tack'
additional.ui <- actionButton("submit", "Submit")
organize_inputs(ui.inputs, tack = additional.ui, columns = 3)

# Example 4: Handling a case with more inputs than grid cells
many.inputs <- tagList(replicate(10, textInput("input", "Input")))
organize_inputs(many.inputs, columns = 3) # Creates more than one row
```

---

pbmc3k_sce *Sample SingleCellExperiment Object*

Description

A sample SingleCellExperiment object created from the pbmc3k dataset for testing and demonstration purposes. The dataset contains 500 cells with gene expression data mapped to Ensembl gene IDs and formatted with cellNexus-compatible metadata structure.

Format

An object of class `SingleCellExperiment` with:

assays Gene expression matrix with Ensembl gene IDs as rownames

colData Cell metadata including sample_id, cell_type_unified_ensemble, nCount_RNA, etc.

metadata List containing 'data' field with cellNexus-formatted metadata including:

- cell_id: Unique cell identifier
- sample_id: Sample identifier
- cell_type_unified_ensemble: Cell type annotation
- nCount_RNA: Number of RNA molecules per cell
- ident: Seurat cluster identity
- dataset_id: Dataset identifier
- file_id_cellNexus_single_cell: Generated file ID for cellNexus
- atlas_id: Atlas identifier with date

Details

See `dev/create_pbmc3k_sce.R` for the complete creation script.

Source

Created from pbmc3k dataset (SeuratData package)

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
data(pbmc3k_sce)
pbmc3k_sce
# Access metadata
S4Vectors::metadata(pbmc3k_sce)$data
```

SAMPLE_DATABASE_URL *URL pointing to the sample metadata file, which is smaller and for test, demonstration, and vignette purposes only*

Description

URL pointing to the sample metadata file, which is smaller and for test, demonstration, and vignette purposes only

Usage

```
SAMPLE_DATABASE_URL
```

Format

An object of class `character` of length 1.

Value

Character scalar consisting of the URL/URLs

Source

Shen et al.,2026

References

Shen, M., Y. Gao, N. Liu, D. Bhuva, M. Milton, J. Henao, J. Andrews, E. Yang, C. Zhan, N. Liu, S. Si, J. W. Hutchison, M. H. Shakeel, M. Morgan, A. T. Papenfuss, J. Iskander, J. M. Polo, and S. Mangiola. "cellNexus: Quality control, annotation, aggregation and analytical layers for the Human Cell Atlas data." bioRxiv (2026). doi:10.64898/2026.04.14.718336.

Examples

```
get_metadata(cloud_metadata = SAMPLE_DATABASE_URL, cache_directory = tempdir())
```

`ui_choices` *Pre-computed UI Choices for Interface App*

Description

A list of unique values for each filterable column used in the cellNexus Shiny interface app. Pre-computing these choices avoids slow metadata queries when the app starts.

Usage

```
data(ui_choices)
```

Format

A named list where each element contains unique values for a column:

cell_type_unified_ensemble Character vector of unified cell type labels

cell_type Character vector of original cell type labels

alive Logical values for cell viability

scDblFinder.class Character vector of doublet classification results

is_immune Logical values for immune cell classification

empty_droplet Logical values for empty droplet detection

development_stage Character vector of developmental stages

disease Character vector of disease states

self_reported_ethnicity Character vector of ethnicity labels

sex Character vector of sex labels

tissue Character vector of tissue types

Details

See `dev/generate_ui_choices.R` for the creation script. Run this script to regenerate the choices when metadata columns change.

Source

Generated from cellNexus metadata

Index

∗ **datasets**
pbmc3k_sce, 20
SAMPLE_DATABASE_URL, 21
ui_choices, 22

cellNexus *(*cellNexus-package*)*, 3
cellNexus-package, 3
create_interface_app, 4

get_atlas_versions, 5
get_cell_communication_strength, 6
get_counts_per_million, 7
get_default_cache_dir, 8
get_metadata, *3*, 9
get_metadata(), *5*, *12*, *14, 15*, *17*
get_metadata_url, 11
get_pseudobulk, *3*, 12
get_seurat, *3*, 14
get_single_cell_experiment, *3*, *14*, 15, *17*
get_single_cell_experiment(), *9*, *14*
get_SingleCellExperiment, 17

keep_quality_cells, 18

organize_inputs, 19

pbmc3k_sce, 20

SAMPLE_DATABASE_URL, 21
SingleCellExperiment::SingleCellExperiment, *9*, *15*, *17*
SummarizedExperiment::SummarizedExperiment, *12*

ui_choices, 22