Package ‘chevreulProcess’

June 4, 2026

Type Package

Title Tools for managing SingleCellExperiment objects as projects

Version 1.5.0

Description Tools for analyzing SingleCellExperiment objects as projects. for
input into the chevreulShiny app downstream.
Includes functions for analysis of single cell RNA sequencing data.
Supported by NIH grants R01CA137124 and R01EY026661 to David Cobrinik.

License MIT + file LICENSE

URL `https://github.com/whtns/chevreulProcess`,
`https://whtns.github.io/chevreulProcess/`

Date 2024-03-24

BugReports `https://github.com/cobriniklab/chevreulProcess/issues`

Depends R (>= 4.5.0), SingleCellExperiment, scater

Imports batchelor, bluster, circlize, cluster, DBI, dplyr,
EnsDb.Hsapiens.v86, ensembldb, fs, GenomicFeatures, glue,
megadepth, methods, purrr, RSQLite, S4Vectors, scran, scuttle,
stringr, tibble, tidyr, tidyselect, utils

Suggests BiocStyle, knitr, RefManageR, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

LazyData false

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

biocViews Coverage, RNASeq, Sequencing, Visualization, GeneExpression,
Transcription, SingleCell, Transcriptomics, Normalization,
Preprocessing, QualityControl, DimensionReduction, DataImport

Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/chevreulProcess

git_branch devel

git_last_commit 3109745

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Author Kevin Stachelek [aut, cre] (ORCID:
<https://orcid.org/0000-0003-2085-695X>),
Bhavana Bhat [aut]

Maintainer Kevin Stachelek <kevin.stachelek@gmail.com>

Contents

`chevreulProcess-package` *chevreulProcess: Tools for managing SingleCellExperiment objects as projects*

Description

Tools analyzing SingleCellExperiment objects as projects. for input into the Chevreul app downstream. Includes functions for analysis of single cell RNA sequencing data. Supported by NIH grants R01CA137124 and R01EY026661 to David Cobrinik.

Author(s)

Maintainer: Kevin Stachelek `<kevin.stachelek@gmail.com>` (ORCID)

Authors:

- Bhavana Bhat `<bhavanabhat29@gmail.com>`

See Also

Useful links:

- `https://github.com/whtns/chevreulProcess`
- `https://whtns.github.io/chevreulProcess/`
- Report bugs at `https://github.com/cobriniklab/chevreulProcess/issues`

`add_percent_mito` *Annotate percent mitochondrial reads per cell*

Description

Add a Percentage of Mitochondrial Read Count Categorical Variable to the Object (based on nCount_RNA)

Usage

```
add_percent_mito(object, experiment = "gene")
```

Arguments

`object`	A object
`experiment`	gene

Value

a single cell object with cell metadata column containing mitochondrial percentage

---

`annotate_cell_cycle` *Annotate Cell Cycle*

Description

Annotate Cell Cycle for Gene and Transcript SingleCellExperiment Objects

Usage

```
annotate_cell_cycle(object)
```

Arguments

`object`	A SingleCellExperiment object

Value

a SingleCellExperiment object

`append_to_project_db` *Update a database of chevreul projects*

Description

Append projects to database

Usage

```
append_to_project_db(
  new_project_path,
  cache_location = "~/.cache/chevreul",
  sqlite_db = "single-cell-projects.db",
  verbose = TRUE
)
```

Arguments

`new_project_path`	new project path
`cache_location`	Path to cache "~/.cache/chevreul"
`sqlite_db`	sqlite db
`verbose`	print messages

Value

a sqlite database with SingleCellExperiment objects

`build_bigwig_db` *Create a database of bigwigfiles*

Description

Create a sqlite database of bigwig files matching cell ids in objects

Usage

```
build_bigwig_db(bam_files, bigwig_db = "~/.cache/chevreul/bw-files.db")
```

Arguments

`bam_files`	vector of paths to bam files
`bigwig_db`	bigwig database

Value

a path to a bigwig file sqlite database

`cc.genes.cyclone` *Cyclone cell cycle pairs by symbol*

Description

cell cycle genes with paired expression represented by HGNC symbol

Usage

```
cc.genes.cyclone
```

Format

a list of dataframes with G1, G2, and S gene expression

G1 G1 gene symbols

G2 G2 gene symbols

S S gene symbols ...

Source

cyclone

`clustering_workflow` *Clustering Workflow*

Description

Cluster and Reduce Dimensions of a object

Usage

```
clustering_workflow(
  object,
  excluded_cells,
  resolution = seq(0.2, 1, by = 0.2),
  organism = "human",
  experiment_name = "default_experiment",
  ...
)
```

Arguments

`object`	a SingleCellExperiment object
`excluded_cells`	named list of cells to exclude
`resolution`	resolution(s) to use for clustering cells
`organism`	Organism
`experiment_name`	name of the experiment
`...`	extra args passed to sce_process

Value

a clustered SingleCellExperiment object

`convert_human_sce_to_mouse` *Convert SingleCellExperiment Objects from Human to Mouse*

Description

Convert SingleCellExperiment Objects from Human to Mouse

Usage

```
convert_human_sce_to_mouse(object, ...)
```

Arguments

`object`	Human SingleCellExperiment object
`...`	to be passed to `convert_symbols_by_species`

Value

a SingleCellExperiment object

`convert_symbols_by_species` *Convert gene symbols between mouse and human*

Description

Convert gene symbols between mouse and human

Usage

```
convert_symbols_by_species(src_genes, src_species)
```

Arguments

`src_genes`	Source gene symbol to be converted
`src_species`	Source species

Value

a SingleCellExperiment object

`create_project_db` *Create a database of chevreul projects*

Description

Create a database containing chevreul projects

Usage

```
create_project_db(
  cache_location = "~/.cache/chevreul",
  sqlite_db = "single-cell-projects.db",
  verbose = TRUE
)
```

Arguments

`cache_location`	Path to cache "~/.cache/chevreul"
`sqlite_db`	Database to be created
`verbose`	print messages

Value

a sqlite database with SingleCellExperiment objects

`ensembl_version` *Ensembl version used for build*

Description

Ensembl version used for build

Usage

```
ensembl_version
```

Format

An object of class `character` of length 1.

Source

`http://www.ensembl.org/`

Examples

```
# ensembl_version
```

`find_all_markers` *Find All Markers*

Description

Find all markers at a range of resolutions

Usage

```
find_all_markers(object, group_by = NULL, experiment = "gene", ...)
```

Arguments

`object`	An object.
`group_by`	A metadata variable to group by.
`experiment`	Assay to use, Default "gene".
`...`	extra args passed to stash_marker_features

Value

a SingleCellExperiment object containing marker genes

Examples

```
data("small_example_dataset")
find_all_markers(small_example_dataset, "gene_snn_res.1")
```

`genes_to_transcripts` *Gene Symbols to Ensembl Transcript Ids*

Description

convert hgnc gene symbols to ensembl transcript ids

Usage

```
genes_to_transcripts(symbols)
```

Arguments

`symbols`	character vector of gene symbols

Value

a vector of transcripts

Examples

```
genes_to_transcripts("NRL")
```

`get_colData` *Get cell metadata from a given object*

Description

Get cell metadata

Usage

```
get_colData(object)
```

Arguments

`object`	a SingleCellExperiment object

Value

dataframe containing object metadata

Examples

```
data(small_example_dataset)
get_colData(small_example_dataset)
```

`get_features` *Get feature names*

Description

Get feature names

Usage

```
get_features(object, experiment = "gene")
```

Arguments

`object`	a SingleCellExperiment object
`experiment`	"gene" or "transcript"

Value

variable features from a SingleCellExperiment object

Examples

```
data(small_example_dataset)
get_features(small_example_dataset)
```

`get_feature_types` *Get Feature Types*

Description

Get Feature Types

Usage

```
get_feature_types(object)
```

Arguments

`object`	a SingleCellExperiment object

Value

vector of feature types in an object

Examples

```
data(small_example_dataset)
get_feature_types(small_example_dataset)
```

`get_sce_metadata` *Get object metadata*

Description

Get object metadata

Usage

```
get_sce_metadata(object)
```

Arguments

`object`	a SingleCellExperiment object

Value

variable features from a SingleCellExperiment object

`get_transcripts_from_sce` *Get Transcripts in object*

Description

Get transcript ids in objects for one or more gene of interest

Usage

```
get_transcripts_from_sce(object, gene)
```

Arguments

`object`	A SingleCellExperiment object
`gene`	Gene of interest

Value

transcripts constituting a gene of interest in a SingleCellExperiment object

---

`get_variable_features` *Get variable features*

Description

Get variable features

Usage

```
get_variable_features(object, experiment = "gene")
```

Arguments

`object`	a SingleCellExperiment object
`experiment`	"gene" or "transcript"

Value

variable features from a SingleCellExperiment object

Examples

```
data(small_example_dataset)
get_variable_features(small_example_dataset)
```

`grch38` *Human annotation data*

Description

Human (*Homo sapiens*) annotations based on genome assembly GRCH38 from Ensembl.

Usage

```
grch38
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 76062 rows and 9 columns.

Details

Variables:

- ensgene
- entrez
- symbol
- chr
- start
- end
- strand
- biotype
- description

Source

http://ensembl.org/homo_sapiens

Examples

```
data("grch38")
head(grch38)
```

`grch38_tx2gene` *Human transcripts to genes*

Description

Lookup table for converting Human (*Homo sapiens*) Ensembl transcript IDs to gene IDs based on genome assembly GRCH38 from Ensembl.

Usage

```
grch38_tx2gene
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 277081 rows and 2 columns.

Details

Variables:

- enstxp
- ensgene

Source

http://ensembl.org/homo_sapiens

Examples

```
data(grch38_tx2gene)
head(grch38_tx2gene)
```

`human_to_mouse_homologs` *Gene Homologs Between Human and Mouse*

Description

Homologs drawn from Biomart

Usage

```
human_to_mouse_homologs
```

Format

A data frame with 23188 rows and 2 columns

HGNC.symbol human gene symbols

MGI.symbol mouse gene symbols ...

Source

bioMart

`integrate` *Batch Correct Multiple Single Cell Objects*

Description

Batch Correct Multiple Single Cell Objects

Usage

```
integrate(sce_list, organism = "human", ...)
```

Arguments

`sce_list`	List of two or more SingleCellExperiment objects
`organism`	human or mouse
`...`	extra args passed to sce_reduce_dimensions

Value

an integrated SingleCellExperiment object

`integration_workflow` *Integration Workflow*

Description

Integrate multiple objects and save to file

Usage

```
integration_workflow(
  batches,
  excluded_cells = NULL,
  resolution = seq(0.2, 1, by = 0.2),
  experiment_name = "default_experiment",
  organism = "human",
  ...
)
```

Arguments

`batches`	objects for all batches provided as a list. If named, the resulting integrated object will be identified with corresponding values in 'batch' metadata
`excluded_cells`	named list of cells to exclude
`resolution`	value(s) to control the clustering resolution via `scran::findMarkers`
`experiment_name`	arbitrary name to identify experiment
`organism`	either "human" or "mouse"
`...`	extra args passed to sce_integrate

Value

an integrated SingleCellExperiment object

`load_bigwigs` *Load Bigwigs*

Description

Load a tibble of bigwig file paths by cell id

Usage

```
load_bigwigs(object, bigwig_db = "~/.cache/chevreul/bw-files.db")
```

Arguments

`object`	A object
`bigwig_db`	Sqlite database of bigwig files

Value

a vector of bigwigs file paths

`load_sce_from_proj` *Load SingleCellExperiment Files from a single project path*

Description

Load SingleCellExperiment Files from a single project path

Usage

```
load_sce_from_proj(proj_dir, ...)
```

Arguments

`proj_dir`	project directory
`...`	extra args passed to load_sce_path

Value

a SingleCellExperiment object

`load_sce_path` *Read in Gene and Transcript SingleCellExperiment Objects*

Description

Read in Gene and Transcript SingleCellExperiment Objects

Usage

```
load_sce_path(proj_dir = getwd(), prefix = "unfiltered")
```

Arguments

`proj_dir`	path to project directory
`prefix`	default "unfiltered"

Value

a SingleCellExperiment object

`make_bigwig_db` *Make Bigwig Database*

Description

Make Bigwig Database

Usage

```
make_bigwig_db(
  new_project = NULL,
  cache_location = "~/.cache/chevreul/",
  sqlite_db = "bw-files.db"
)
```

Arguments

`new_project`	Project directory
`cache_location`	Path to cache "~/.cache/chevreul"
`sqlite_db`	sqlite db containing bw files

Value

a sqlite database of bigwig files for cells in a SingleCellExperiment object

---

`merge_small_sces` *Merge Small SingleCellExperiment Objects*

Description

Merge Small SingleCellExperiment Objects

Usage

```
merge_small_sces(..., k.filter = 50)
```

Arguments

`...`	two or more singlecell objects
`k.filter`	minimum cell number for integration

Value

a SingleCellExperiment object

`metadata_from_batch` *Retrieve Metadata from Batch*

Description

Retrieve Metadata from Batch

Usage

```
metadata_from_batch(
  batch,
  projects_dir = "/dataVolume/storage/single_cell_projects",
  db_path = "single-cell-projects.db"
)
```

Arguments

`batch`	batch
`projects_dir`	path to project dir
`db_path`	path to .db file

Value

a tibble with cell level metadata from a SingleCellExperiment object

---

`propagate_spreadsheet_changes` *Propagate Metadata Changes*

Description

Propagate Metadata Changes

Usage

```
propagate_spreadsheet_changes(meta, object)
```

Arguments

`meta`	updated metadata
`object`	a SingleCellExperiment object

Value

a SingleCellExperiment object

Examples

```
data(small_example_dataset)
new_meta <- data.frame(row.names = colnames(small_example_dataset))
new_meta$example <- "example"

propagate_spreadsheet_changes(new_meta, small_example_dataset)
```

`query_experiment` *Query Experiment*

Description

Query Experiment

Usage

```
query_experiment(object, experiment)
```

Arguments

`object`	a SingleCellExperiment object
`experiment`	an experiment name

Value

logical scalar indicating if experiment is present in object

Examples

```
data(small_example_dataset)
query_experiment(small_example_dataset, "gene")
```

`read_project_db` *Read a database of chevreul projects*

Description

Reads database of chevreul projects to a data frame

Usage

```
read_project_db(
  cache_location = "~/.cache/chevreul",
  sqlite_db = "single-cell-projects.db",
  verbose = TRUE
)
```

Arguments

`cache_location`	Path to cache "~/.cache/chevreul"
`sqlite_db`	sqlite db
`verbose`	print messages

Value

a tibble with SingleCellExperiment objects

`record_experiment_data` *Record Experiment Metadata*

Description

Records miscellaneous data

Usage

```
record_experiment_data(
  object,
  experiment_name = "default_experiment",
  organism = "human"
)
```

Arguments

`object`	A object
`experiment_name`	name of the experiment
`organism`	human or mouse

Value

a SingleCellExperiment object

Examples

```
data(small_example_dataset)
record_experiment_data(small_example_dataset)
```

`regress_cell_cycle` *Regress SingleCellExperiment Object by Given Set of Genes*

Description

Regress SingleCellExperiment Object by Given Set of Genes

Usage

```
regress_cell_cycle(object)
```

Arguments

`object`	A object

Value

a SingleCellExperiment object with features regressed

`reintegrate_sce` *Reintegrate (filtered) SingleCellExperiment objects*

Description

This function takes a SCE object and performs the below steps

1. split by batch
2. integrate
3. run integration pipeline and save

Usage

```
reintegrate_sce(object, suffix = "", reduction = "PCA", ...)
```

Arguments

`object`	A SingleCellExperiment objects
`suffix`	to be appended to file saved in output dir
`reduction`	to use default is pca
`...`	extra args passed to sce_integrate

Value

a SingleCellExperiment object

`retrieve_experiment` *Retrieve Assay*

Description

Retrieve Assay

Usage

```
retrieve_experiment(object, experiment)
```

Arguments

`object`	a SingleCellExperiment object
`experiment`	an experiment name

Value

Main or alt experiment in a SingleCellExperiment object

`save_sce` *Save object to /output/sce/_sce.rds*

Description

Save object to /output/sce/_sce.rds

Usage

```
save_sce(object, prefix = "unfiltered", proj_dir = getwd())
```

Arguments

`object`	a SingleCellExperiment object
`prefix`	a prefix for saving
`proj_dir`	path to a project directory

Value

a path to an rds file containing a SingleCellExperiment object

---

`sce_calcn` *Calculate Read Count Metrics for a object*

Description

Recalculate counts/features per cell for a object

Usage

```
sce_calcn(object)
```

Arguments

`object`	A SingleCellExperiment object

Value

a SingleCellExperiment object with nfeatures and ngenes stored in metadata

Examples

```
data(small_example_dataset)
sce_calcn(small_example_dataset)
```

`sce_cluster` *Run Louvain Clustering at Multiple Resolutions*

Description

Run Louvain Clustering at Multiple Resolutions

Usage

```
sce_cluster(
  object = object,
  resolution = 0.6,
  custom_clust = NULL,
  reduction = "PCA",
  algorithm = 1,
  ...
)
```

Arguments

`object`	A SingleCellExperiment objects
`resolution`	Clustering resolution
`custom_clust`	custom cluster
`reduction`	Set dimensional reduction object
`algorithm`	1
`...`	extra args passed to single cell packages

Value

a SingleCellExperiment object with louvain clusters

Examples

```
data(small_example_dataset)
sce_cluster(small_example_dataset)
```

`sce_de` *Run Differential Expression*

Description

Run Differential Expression

Usage

```
sce_de(
  object,
  cluster1,
  cluster2,
  resolution = 0.2,
  diffex_scheme = "louvain",
  featureType = "gene",
  tests = c("t", "wilcox", "bimod")
)
```

Arguments

`object`	a SingleCellExperiment object
`cluster1`	cluster 1
`cluster2`	cluster 2
`resolution`	resolution
`diffex_scheme`	scheme for differential expression
`featureType`	gene or transcript
`tests`	t, wilcox, or bimod

Value

a dataframe with differential expression information

Examples

```
data("tiny_sce")
sce_de(tiny_sce,
colnames(tiny_sce)[1:100],
colnames(tiny_sce)[101:200],
diffex_scheme = "custom")
```

`sce_integrate` *Run SingleCellExperiment Integration*

Description

Run batch correction, followed by:

1. stashing of batches in metadata 'batch'
2. clustering with resolution 0.2 to 2.0 in increments of 0.2
3. saving to <proj_dir>/output/sce/*sce*.rds

Usage

```
sce_integrate(
  sce_list,
  resolution = seq(0.2, 1, by = 0.2),
  suffix = "",
  organism = "human",
  batch_correct = TRUE,
  annotate_cell_cycle = FALSE,
  annotate_percent_mito = FALSE,
  reduction = "corrected",
  ...
)
```

Arguments

`sce_list`	List of objects to be integrated
`resolution`	Range of resolution
`suffix`	a suffix to be appended to a file save in output dir
`organism`	Default "human"
`batch_correct`	whether to integrate by batch correction
`annotate_cell_cycle`	whether to score cell cycle phases
`annotate_percent_mito`	logical scalar whether to annotate mitochondrial percentage
`reduction`	pca, umap, or tsne
`...`	extra args passed to integrate

Value

an integrated SingleCellExperiment object

Examples

```
data("tiny_sce")
tiny_sce |>
splitByCol("bath") |>
sce_integrate(resolution = 0.2, batch_correct = FALSE)
```

`sce_preprocess` *Preprocess Single Cell Object*

Description

Performs standard pre-processing workflow for scRNA-seq data

Usage

```
sce_preprocess(
  object,
  scale = TRUE,
  normalize = TRUE,
  features = NULL,
  legacy_settings = FALSE,
  ...
)
```

Arguments

`object`	Assay to use
`scale`	Perform linear transformation 'Scaling'
`normalize`	Perform normalization
`features`	Identify highly variable features
`legacy_settings`	Use legacy settings
`...`	extra args passed to scaling functions

Value

a preprocessed SingleCellExperiment object

`sce_process` *Run SingleCellExperiment Pipeline*

Description

This functions allows you to preprocess, cluster and reduce dimensions for one SingleCellExperiment object.

Usage

```
sce_process(
  object,
  experiment = "gene",
  resolution = 0.6,
  reduction = "PCA",
  organism = "human",
  process = TRUE,
  ...
)
```

Arguments

`object`	A SingleCellExperiment object
`experiment`	Assay of interest in SingleCellExperiment object
`resolution`	Resolution for clustering cells. Default set to 0.6.
`reduction`	Dimensional reduction object
`organism`	Organism
`process`	whether to run dimensional reduction and clustering
`...`	extra parameters passed to internal functions

Value

a processed SingleCellExperiment object

Examples

```
data(tiny_sce)
sce_process(tiny_sce, process = FALSE)
```

`sce_reduce_dimensions` *Dimensional Reduction*

Description

Run PCA, TSNE and UMAP on a singlecell objects perplexity should not be bigger than 3 * perplexity < nrow(X) - 1, see details for interpretation

Usage

```
sce_reduce_dimensions(object, experiment = "gene", ...)
```

Arguments

`object`	A SingleCellExperiment object
`experiment`	Experiment of interest to be processed
`...`	Extra parameters passed to sce_reduce_dimensions

Value

a SingleCellExperiment object with embeddings

`set_colData` *Set cell metadata*

Description

Set cell metadata from a given object

Usage

```
set_colData(object, meta)
```

Arguments

`object`	a SingleCellExperiment object
`meta`	a dataframe containing object metadata

Value

a SingleCellExperiment object with new colData

Examples

```
data(small_example_dataset)
new_meta <- data.frame(row.names = colnames(small_example_dataset))
new_meta$example <- "example"
set_colData(small_example_dataset, new_meta)
```

`set_feature_type` *Set Feature Types*

Description

Set Feature Types

Usage

```
set_feature_type(object, feature_type)
```

Arguments

`object`	a SingleCellExperiment object
`feature_type`	feature type

Value

a SingleCellExperiment object with assigned feature type

Examples

```
data(small_example_dataset)
set_feature_type(small_example_dataset, "transcript")
```

`small_example_dataset` *Small example SingleCellExperiment*

Description

created with scuttle::mockSCE

Usage

```
small_example_dataset
```

Format

An SCE with 200 cells and 1000 genes

Source

scuttle::mockSCE

`splitByCol` *Split SingleCellExperiment by colData variable*

Description

Split SingleCellExperiment by colData variable

Usage

```
splitByCol(x, f = "batch")
```

Arguments

`x`	SingleCellExperiment object
`f`	colData variable as a string

Value

a list of singlecellexperiments name by colData value

Examples

```
data(small_example_dataset)
splitByCol(small_example_dataset, "batch")
```

`stash_marker_features` *Stash Marker Genes in a SingleCellExperiment Object*

Description

Marker Genes will be stored in object metadata as `markers`

Usage

```
stash_marker_features(
  object,
  group_by,
  experiment = "gene",
  top_n = 200,
  p_val_cutoff = 0.5
)
```

Arguments

`object`	A object
`group_by`	A metadata variable to group by
`experiment`	An experiment to use
`top_n`	Use top n genes, Default 200
`p_val_cutoff`	p value cut-off, Default value is "0.5"

Value

a SingleCellExperiment object with marker genes

`subset_by_colData` *Subset by new colData*

Description

Subset the object using new colData

Usage

```
subset_by_colData(colData_path, object)
```

Arguments

`colData_path`	Path to new colData
`object`	A object

Value

a SingleCellExperiment object

`tiny_sce` *Tiny example SingleCellExperiment*

Description

subset to only NRL from chevreuldata::human_gene_transcript_sce()

Usage

```
tiny_sce
```

Format

An SCE with only expression of NRL gene and NRL transripts

Source

chevreuldata::human_gene_transcript_sce()

`transcripts_to_genes` *Ensembl Transcript Ids to Gene Symbols*

Description

Convert ensembl transcript ids to hgnc gene symbols

Usage

```
transcripts_to_genes(transcripts)
```

Arguments

`transcripts`	human transcripts

Value

a vector of gene symbols

Examples

```
NRL_transcripts_hs <-
c("ENST00000359842", "ENST00000470566", "ENST00000465764")

transcripts_to_genes(transcripts = NRL_transcripts_hs)
```

`update_project_db` *Update a database of chevreul projects*

Description

Add new/update existing projects to the database by recursing fully

Usage

```
update_project_db(
  projects_dir = NULL,
  cache_location = "~/.cache/chevreul",
  sqlite_db = "single-cell-projects.db",
  verbose = TRUE
)
```

Arguments

`projects_dir`	The project directory to be updated
`cache_location`	Path to cache "~/.cache/chevreul"
`sqlite_db`	sqlite db
`verbose`	print messages

Value

a sqlite database with SingleCellExperiment objects

Index