

Package: scooter (via r-universe)

October 7, 2026

Title Streamlined scRNA-Seq Analysis Pipeline

Version 0.0.0.9005

Description Streamlined scRNA-Seq analysis pipeline.

License MIT + file LICENSE

Depends R (>= 4.4), Seurat (>= 5.3.1)

Imports cowplot, data.table, dplyr, eulerr, future, ggplot2, ggsci,
glue, irlba, lifecycle, Matrix, matrixStats, methods, pheatmap,
purrr, qs2, RColorBrewer, readr, rlang (>= 1.0.0), Rtsne,
scales, SeuratObject (>= 5.0.0), stringr, tibble, tidyr,
UpSetR, uwot (>= 0.2.1)

Suggests roxygen2, testthat

Config/roxygen2/version 8.1.0

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-
dev python3 libx11-dev zlib1g-dev libclang-dev

Repository <https://igordot.r-universe.dev>

Date/Publication 2026-10-07 20:56:57 UTC

RemoteUrl <https://github.com/igordot/scooter>

RemoteRef HEAD

RemoteSha 5514bdf4ca9a74e6a4da4d7561015118657632c

Contents

add_gene_class_percent	3
add_seurat_assay	4
as_data_frame_seurat	4
calculate_cluster_expression	5
calculate_cluster_markers	6
calculate_cluster_stats	6
calculate_clusters	7

check_identity_column	8
cluster_seurat_object	9
create_color_vect	10
create_seurat_object	10
differential_expression_per_cluster	12
dr_plot_width	13
filter_cells	14
geneset_score	15
get_color_scheme	16
get_dr_point_size	16
get_test_counts_matrix	17
import_mtx	17
initialize_seurat_object	18
integrate_layers	19
integrate_seurat_object	20
merge_metadata	22
merge_seurat_objects	22
normalize_counts	23
plot_cluster_markers_heatmap	24
plot_cluster_markers_top	25
plot_clusters	25
plot_distribution	26
plot_dr_feature	27
plot_dr_group	28
plot_dr_umap_clusters	29
plot_metrics_correlations	29
plot_metrics_distribution	30
plot_var_genes_euler	31
plot_var_genes_upset	32
profile_object_size	32
read_counts_file	33
resolve_seurat_object	34
run_dr	34
run_pca	35
run_tsne	36
run_umap	37
save_counts	38
save_dr_plot	39
save_metadata	40
set_identity	40
split_layers_by_batch	41
transfer_labels	42
variable_features_by_batch	43
write_message	44

`add_gene_class_percent`

Add mitochondrial, ribosomal, and hemoglobin percentages to a Seurat object.

Description

Detected by gene symbol prefix, case-insensitively so the same patterns match both human (MT-, RPS/RPL, HBA/HBB) and mouse (mt-, Rps/Rpl, Hba-a1/Hbb-bs) naming. RPS6K* (ribosomal protein S6 kinase) is excluded from pct_ribo - it matches the RPS prefix but is a signaling kinase, not a ribosomal protein.

Usage

```
add_gene_class_percent(x)
```

Arguments

x A Seurat object.

Details

High expression levels of mitochondrial genes could be an indicator of poor sample quality, leading to a high fraction of apoptotic or lysing cells (see reference below).

pct_ribo is normally 15-45%; the exact fraction varies by cell type and overall cell health. "Although both ribosomal proteins and ribosomal RNA (rRNA) make up the ribosome complex, ribosomal protein transcripts are not equivalent to ribosomal RNA (rRNA). Ribosomal protein transcript detection will not necessarily correlate with either rRNA or mitochondrial transcripts." (see reference below)

High pct_hb is a contamination signal in PBMC and most solid tissue: per-sample pct_hb flags incomplete red blood cell lysis in PBMC preps, or inadequate perfusion in solid tissue.

Value

Seurat object with pct_mito, pct_ribo, and pct_hb metadata columns added.

References

pct_mito: <https://kb.10xgenomics.com/s/article/360001086611-Why-do-I-see-a-high-level-of-mitochondr>

pct_ribo: <https://kb.10xgenomics.com/s/article/218169723-What-fraction-of-reads-map-to-ribosomal-pr>

add_seurat_assay	<i>Add assay to Seurat object.</i>
------------------	------------------------------------

Description

Add assay to Seurat object.

Usage

```
add_seurat_assay(x, assay, counts_matrix, log_file = NULL)
```

Arguments

x	Seurat object.
assay	Seurat assay to add the matrix to.
counts_matrix	Raw counts matrix.
log_file	Filename for the log file.

Value

Seurat object of cells found in both the existing object and new data.

as_data_frame_seurat	<i>Function to extract data from Seurat object.</i>
----------------------	---

Description

Function to extract data from Seurat object.

Usage

```
as_data_frame_seurat(  
  x,  
  assay = NULL,  
  slot = NULL,  
  features = NULL,  
  reduction = NULL,  
  metadata = TRUE  
)
```

Arguments

x	A Seurat object.
assay	Assay such as RNA.
slot	Slot such as counts. Default is scale.data.
features	Features from assay.
reduction	Character vector of reduction types.
metadata	Boolean. To grab metadata or not

Value

A metadata file merged on cell identifiers.

calculate_cluster_expression
Per-cluster average expression (non-log space).

Description

Writes expression-mean-<label>.csv, and additionally expression-mean-<label>-per-sample.csv when x has more than one sample.

Usage

```
calculate_cluster_expression(x, label)
```

Arguments

x	Seurat object with an identity set, such as via set_identity() .
label	Suffix for the output filenames.

Value

x, invisibly. Called for its file output.

```
calculate_cluster_markers
```

Calculate cluster markers (versus all other clusters, or pairwise) and plot them.

Description

test, forwarded to `Seurat::FindAllMarkers()/Seurat::FindMarkers()`: "wilcox" (Wilcoxon rank sum, the default), "roc" (classification power from 0 = random to 1 = perfect), "bimod" (McDavid et al. 2013), "tobit" (Trapnell et al. 2014), or "MAST" (Finak et al. 2015).

Usage

```
calculate_cluster_markers(x, label, test, pairwise = FALSE)
```

Arguments

x	Seurat object with an identity set.
label	Suffix for the output filenames.
test	Statistical test - see above.
pairwise	Compare clusters pairwise instead of each cluster against all others - see above.

Details

pairwise = TRUE compares each cluster against each other cluster individually (rather than each cluster against all others combined), keeping only genes significant in every comparison - local and global markers both, at the cost of being much slower for many clusters.

Value

x, invisibly. Called for its file and plot output.

```
calculate_cluster_stats
```

Per-cluster cell counts and a joined metadata+embeddings table.

Description

Writes metadata-<label>.csv.gz (cell metadata joined with the tSNE/UMAP embeddings, skipped if neither reduction is present) and summary-<label>.csv (cell counts and percentages per cluster, split further by sample into summary-<label>-per-sample.csv when that differs).

Usage

```
calculate_cluster_stats(x, label)
```

Arguments

x	Seurat object with an identity set, such as via <code>set_identity()</code> .
label	Suffix for the output filenames.

Value

x, invisibly. Called for its file output.

calculate_clusters	<i>Identify clusters of cells by graph-based clustering.</i>
--------------------	--

Description

Builds a shared nearest neighbor (SNN) graph from reduced dimensions and identifies clusters over a range of resolutions.

Usage

```
calculate_clusters(x, ...)

## S3 method for class 'Seurat'
calculate_clusters(
  x,
  assay,
  reduction,
  num_dim,
  num_neighbors = 20,
  res = NULL,
  algorithm = 4,
  log_file = NULL,
  max_clusters = length(get_color_scheme("clusters")),
  ...
)
```

Arguments

x	A matrix of cell embeddings (cells as rows) or a Seurat object.
...	Arguments passed to the individual methods.
assay	Assay logged as the one this clustering run is over (Seurat method). Purely for the log message - clustering itself runs on <code>Embeddings(x, reduction = reduction)</code> , not on assay data, so nothing here reads it back off the object. No default: <code>DefaultAssay(x)</code> is mutable state that need not match the assay reduction actually came from, so the caller states it explicitly rather than have the log silently report the wrong one.
reduction	Reduction to take the cell embeddings from (Seurat method). No default - always state it explicitly.

num_dim	Number of dimensions to use.
num_neighbors	Number of neighbors (k.param) used to build the SNN graph.
res	Clustering resolution(s). If NULL, a range of resolutions is used.
algorithm	Clustering algorithm: 1 = Louvain, 2 = Louvain with multilevel refinement, 3 = SLM, 4 = Leiden (the default). See <code>Seurat::FindClusters()</code> .
log_file	Filename for the log file.
max_clusters	Drop resolutions yielding this many clusters or more. Defaults to the length of the cluster color scheme, since resolutions that exceed it cannot be plotted (Seurat method).

Details

For a Seurat object, the cluster assignments are added to the object metadata as `res.<resolution>` columns, the labels are converted to the C01, C02, ... convention (1-based, zero-padded, and C-prefixed to block downstream numeric coercion), and resolutions that do not yield any new clusters are dropped.

Value

Cluster assignments as a data frame (default method) or the Seurat object with the cluster assignments added to its metadata (Seurat method).

`check_identity_column` *Check identity of the Seurat object.*

Description

Check identity of the Seurat object.

Usage

```
check_identity_column(x, identity_column)
```

Arguments

`x` A Seurat object.

`identity_column` The name of the identity column to pull from object metadata.

Value

The name of the identity column, potentially corrected if resolution.

cluster_seurat_object *Reduce dimensions, cluster, and plot the result.*

Description

The clustering step of the pipeline in one call: run tSNE and UMAP on an existing "pca" reduction ([run_tsne\(\)](#), [run_umap\(\)](#)), build the SNN graph and cluster over a range of resolutions ([calculate_clusters\(\)](#)), and plot each reduction by cluster.

Usage

```
cluster_seurat_object(
  x,
  assay = "RNA",
  reduction,
  num_dim,
  num_neighbors = 30,
  metadata_file = NULL,
  log_file = NULL
)
```

Arguments

x	Seurat object with reduction computed, such as the output of create_seurat_object() (which only ever produces "pca") or integrate_seurat_object() (which also produces one named after its integration method: "cca", "rpca", or "harmony").
assay	Assay to attribute the tSNE/UMAP reductions to. create_seurat_object() creates its own assay, so it needs no such parameter. This function creates none: it only calls run_tsne()/run_umap() , which read from an existing assay. So assay is explicit here, rather than read back off <code>DefaultAssay(x)</code> .
reduction	Reduction to compute tSNE/UMAP from and to cluster on - "pca" after create_seurat_object() , or the integration method's own name ("cca"/"rpca"/"harmony") after integrate_seurat_object() to cluster on the batch-corrected embedding instead of the pre-correction one. No default: which reduction is the intended one for clustering differs by which pipeline branch got you here, so every caller states it explicitly rather than risk silently picking the wrong one.
num_dim	Principal components to use, bounded to 5-50.
num_neighbors	Neighbors for UMAP and for the SNN graph.
metadata_file	Path to write the cell metadata and embeddings to, via save_metadata() . NULL writes nothing.
log_file	Filename for the log file.

Details

This function computes both reductions before clustering, since the cluster plots draw on them. `run_tsne()` and `run_umap()` already write their own sample-colored scatter to the working directory, so this function skips plotting "by sample" again.

The per-resolution cluster plots also write unconditionally, into a `clusters-resolutions/` subdirectory. This matches every other verb's standard-output convention.

Value

A Seurat object with "tsne" and "umap" reductions and one `res.<x>` column per retained clustering resolution.

<code>create_color_vect</code>	<i>Function to create a color vector.</i>
--------------------------------	---

Description

Function to create a color vector.

Usage

```
create_color_vect(seurat_obj, group = "orig.ident")
```

Arguments

<code>seurat_obj</code>	A Seurat object.
<code>group</code>	Assay such as RNA.

Value

A vector of colors.

<code>create_seurat_object</code>	<i>Create a usable Seurat object from a sample.</i>
-----------------------------------	---

Description

Everything needed to take a sample from a counts file to an object ready for clustering, in one call.

Usage

```
create_seurat_object(
  path,
  sample_name,
  num_mads = 3,
  min_genes = 500,
  max_genes = NULL,
  min_counts = NULL,
  max_counts = NULL,
  max_mt = 10,
  min_cells = 100,
  normalization_method = "log",
  num_variable_genes = 3000,
  num_pcs = 50,
  num_dim = 30,
  log_file = NULL
)
```

Arguments

path	Path to a 10x directory or a flat counts file.
sample_name	Sample/library name, used as the cell ID prefix.
num_mads, min_genes, max_genes, min_counts, max_counts, max_mt	Quality cutoffs, passed to filter_cells() .
min_cells	Minimum number of cells that must survive filtering.
normalization_method	Normalization method, passed to normalize_counts() (as its own method argument): "log" or "sct".
num_variable_genes	Number of variable features.
num_pcs	Principal components to compute. Reduced automatically when the object is too small to support that many.
num_dim	Number of dimensions for the first-pass tSNE/UMAP, capped at num_pcs. NULL or a length-zero vector falls back to the default, same as omitting it.
log_file	Filename for the log file.

Details

The function runs these steps:

1. Read the counts ([read_counts_file\(\)](#)).
2. Build the object ([initialize_seurat_object\(\)](#)).
3. Attach any antibody capture data as an "ADT" assay ([add_seurat_assay\(\)](#)).
4. Apply the quality cutoffs ([filter_cells\(\)](#)).
5. Normalize, select variable features, and scale ([normalize_counts\(\)](#), which does all three for either method).

6. Run PCA ([run_pca\(\)](#)).
7. Compute a first-pass tSNE/UMAP ([run_tsne\(\)](#), [run_umap\(\)](#)).

This function attaches the ADT assay *before* filtering, on purpose. [add_seurat_assay\(\)](#) restricts the object to the barcodes both matrices share. Attaching ADT after filtering would apply the cutoffs to a different set of cells.

PCA, and the tSNE/UMAP computed from it, runs on whichever assay normalization left active. SCTransform() creates and activates "SCT". The log path leaves "RNA" active. So `dr_assay` is decided directly from `normalization_method`, not read back off the object. Reading it back off `DefaultAssay()` would reintroduce the implicit dependency an explicit assay argument exists to avoid.

The tSNE/UMAP computed here are a first-pass look, capped at `num_dim` dimensions. [cluster_seurat_object\(\)](#) recomputes both once a clustering resolution's own `num_dim` is chosen. So this pair is not the final one used for the actual clustering plots.

Value

A Seurat object with normalized data, variable features, scaled data, and "pca", "tsne", and "umap" reductions — ready for [cluster_seurat_object\(\)](#). Also writes the variable-feature, PCA, tSNE, and UMAP diagnostic plots (via [normalize_counts\(\)](#), [run_pca\(\)](#), [run_tsne\(\)](#), and [run_umap\(\)](#)) and the metadata+embeddings table (via [save_metadata\(\)](#), as `metadata.csv.gz`) to the working directory.

differential_expression_per_cluster

Calculate differentially expressed genes within each subpopulation/cluster

Description

Calculate differentially expressed genes within each subpopulation/cluster

Usage

```
differential_expression_per_cluster(
  x,
  cluster_column,
  group_column,
  test = "wilcox",
  out_path = ".",
  write = TRUE,
  log_file = NULL
)
```

Arguments

x	A Seurat object.
cluster_column	Metadata column specifying the groups to split by.
group_column	Metadata column specifying the groups for differential expressin within each split.
test	Statistical method to use.
out_path	Output path.
write	Boolean to save results to disk.
log_file	log file.

Value

.

dr_plot_width	<i>Width for a reduction plot, widened so many legend labels stay legible.</i>
---------------	--

Description

A label-count estimate made without building the plot. `plot_dr_group()` now sizes itself from the actual rendered legend instead (see `save_dr_plot()`), so this has no callers left in the package - kept as a cheap standalone estimate for callers that don't have a plot object yet.

Usage

```
dr_plot_width(x, group_by = "orig.ident")
```

Arguments

x	Seurat object, or the number of legend labels.
group_by	Metadata column whose distinct values become the plot's legend. Ignored if x is already a number.

Value

Numeric width in inches.

filter_cells	<i>Filter cells based on the number of genes, counts, and mitochondrial reads.</i>
--------------	--

Description

Filtering runs in two stages, and the log records how many cells each one removed.

Usage

```
filter_cells(
  x,
  num_mads = 3,
  min_genes = 500,
  max_genes = NULL,
  min_counts = NULL,
  max_counts = NULL,
  max_mt = 10,
  min_cells = 50,
  log_file = NULL
)
```

Arguments

x	A tibble with metadata (default method) or a Seurat object (Seurat method). Must have detected_genes/total_counts columns, as added by initialize_seurat_object() .
num_mads	Median absolute deviations for the outlier stage. NULL skips the stage.
min_genes	Minimum number of genes per cell. NULL for no cutoff.
max_genes	Maximum number of genes per cell. NULL for no cutoff.
min_counts	Minimum number of counts per cell. NULL for no cutoff.
max_counts	Maximum number of counts per cell. NULL for no cutoff.
max_mt	Maximum percentage of mitochondrial reads per cell. NULL for no cutoff.
min_cells	Minimum number of cells that must survive filtering. Below this, filtering stop()s rather than handing back an object too small to analyze meaningfully. NULL skips the check.
log_file	Log file.

Details

1. **Outlier removal.** Cells further than num_mads median absolute deviations from the median number of genes or counts are dropped. The median and MAD are computed on the log scale and the bounds back-transformed, which suits these right-skewed counts. num_mads = NULL skips it.

- 2. **Fixed cutoffs.** min_genes / max_genes / min_counts / max_counts are applied to whatever survived stage 1, then max_mt. Each is skipped when NULL, so out of the box only min_genes (500) and max_mt (10) do anything.

Applying the fixed cutoffs second makes them a quality floor the data-driven bounds cannot undercut: a wide or contaminated distribution can put the MAD bound below 500 genes, and those cells are still removed. All cutoffs are inclusive.

Genes that are no longer detected in any of the retained cells are dropped, since filtering cells leaves them as all-zero rows.

Value

Filtered data. The Seurat method also writes a pre-filtering QC violin plot (via `plot_metrics_distribution()`) and the unfiltered metadata (via `save_metadata()`, as metadata-unfiltered.csv.gz), plus the post-filtering QC plots (via `plot_metrics_distribution()` and `plot_metrics_correlations()`) to the working directory.

geneset_score	<i>Get geneset scores.</i>
---------------	----------------------------

Description

Get geneset scores.

Usage

`geneset_score(module_tbl, counts_raw, min_cpm = 0, limit_pct = 1)`

Arguments

module_tbl	geneset table.
counts_raw	Raw counts
min_cpm	.
limit_pct	.

Value

.

get_color_scheme	<i>Determine the color scheme.</i>
------------------	------------------------------------

Description

Determine the color scheme. Can be specified for samples or clusters to avoid confusion.

Usage

```
get_color_scheme(type = "clusters")
```

Arguments

type	Type of scheme ("samples" or "clusters").
------	---

Value

A vector of colors.

get_dr_point_size	<i>Determine the point size for reduced dimensions scatter plots (smaller for larger datasets).</i>
-------------------	---

Description

Determine the point size for reduced dimensions scatter plots (smaller for larger datasets).

Usage

```
get_dr_point_size(x)
```

Arguments

x	Number of cells (points on the plot), a Seurat object, or a data frame with one row per cell.
---	---

Value

Numeric point size.

`get_test_counts_matrix`*Get an example counts matrix.*

Description

Get a small matrix of raw counts from the PBMC dataset.

Usage

```
get_test_counts_matrix()
```

Value

A matrix of raw counts.

Examples

```
pbmc_mat <- get_test_counts_matrix()
```

`import_mtx`*Read in 10x Genomics Cell Ranger Matrix Market format data.*

Description

Read in 10x Genomics Cell Ranger Matrix Market format data.

Usage

```
import_mtx(data_path, gene_column = 2, log_file = NULL)
```

Arguments

<code>data_path</code>	Path to directory that holds the files output from 10x.
<code>gene_column</code>	The column with the gene names.
<code>log_file</code>	Filename for the log file.

Value

Named list of matrices. One matrix for each data type as specified in the third column of the features.tsv file. As of Oct 3rd 2019, the two options are Gene Expression and Antibody Capture

`initialize_seurat_object`*Create a new Seurat object from a matrix.*

Description

For an RNA assay the per-cell QC metrics are added at the same time: `pct_mito/pct_ribo/pct_hb` (via `add_gene_class_percent()`), plus `detected_genes` and `total_counts` as clearer aliases of `nFeature_RNA/nCount_RNA`, and `sample_name` as a clearer alias of `orig.ident`. All six are metadata columns as soon as the object exists, so `plot_metrics_distribution()` (which takes its metrics explicitly, no default) and `plot_metrics_correlations()` can use any of them without further preparation.

Usage

```
initialize_seurat_object(  
  counts_matrix,  
  assay = "RNA",  
  min_cells = 1,  
  min_genes = 1,  
  project = "proj",  
  log_file = NULL  
)
```

Arguments

<code>counts_matrix</code>	A matrix of raw counts.
<code>assay</code>	Seurat assay to add the data to.
<code>min_cells</code>	Include genes/features detected in at least this many cells.
<code>min_genes</code>	Include cells where at least this many genes/features are detected.
<code>project</code>	Project name for Seurat object.
<code>log_file</code>	Filename for the logfile.

Value

Seurat object.

integrate_layers	<i>Integrate the layers of a Seurat object.</i>
------------------	---

Description

Runs the Seurat 5 `IntegrateLayers()` workflow. It adds what calling `IntegrateLayers()` directly leaves for the caller to handle:

Usage

```
integrate_layers(
  x,
  assay = "RNA",
  int_reduction = "cca",
  num_dim,
  k_anchor = 10,
  k_weight = 100,
  log_file = NULL,
  ...
)
```

Arguments

<code>x</code>	Seurat object with split layers and a "pca" reduction.
<code>assay</code>	Assay to integrate.
<code>int_reduction</code>	Integration method: "cca", "rpca", or "harmony".
<code>num_dim</code>	Number of dimensions to use (5-50). For "cca"/"rpca", this value is passed as <code>dims = 1:num_dim</code> to <code>CCAIntegration()/RPCAIntegration()</code> , which both take a real <code>dims</code> argument. <code>HarmonyIntegration()</code> has no <code>dims</code> parameter. It runs on the full "pca" embedding as-is, through <code>Embeddings(orig)</code> , unsliced. So for <code>int_reduction = "harmony"</code> , <code>num_dim</code> only takes effect indirectly, through however many components <code>orig.reduction("pca")</code> already has. The normal <code>integrate_seurat_object()</code> pipeline is unaffected: <code>split_layers_by_batch()</code> already computes that "pca" reduction with exactly <code>num_dim</code> components. A direct call is different. If <code>x</code> 's "pca" reduction already has more components than <code>num_dim</code> , then <code>num_dim</code> has no effect on the harmony path.
<code>k_anchor</code>	Neighbors to use when picking anchors. Ignored by "harmony". A value larger than Seurat's default of 5 strengthens the alignment.
<code>k_weight</code>	Neighbors to use when weighting the corrections. Must be smaller than the number of cells in the smallest batch. Reduced to 25 when the smallest batch has fewer than 100 cells. Batches smaller than 25 cells are an error.
<code>log_file</code>	Filename for the log file.
<code>...</code>	Extra arguments for whichever method <code>IntegrateLayers()</code> dispatches to - e.g. <code>theta/lambda/sigma/tau/nclust</code> for "harmony", or <code>k.filter/sample.tree/normalization.method</code> for "cca"/"rpca". An unrecognized name errors (via

`rlang::check_dots_used()` rather than being silently dropped by the method function's own

Details

- Clamps `k_weight` to fit the smallest batch. Seurat's own default of 100 errors outright below that size. See `k_weight` below.
- Excludes `dims/k.anchor` for "harmony". `HarmonyIntegration()` silently ignores both instead of erroring, so passing them would rely on that silent drop. See `num_dim/k_anchor` below.
- Forces `future::plan(future::sequential)`. Benchmarks show multisession gives no gain here, and is a documented liability at this scale elsewhere in this package.
- Rejoins the split layers with `JoinLayers()` afterward. `IntegrateLayers()` itself leaves this undone.

Unlike the older anchor workflow, this function creates no "integrated" assay. Clustering and visualization use the corrected embedding directly, instead of re-running PCA. This function expects `split_layers_by_batch()` to have run first. It is the one step of `integrate_seurat_object()` that calls `IntegrateLayers()` directly.

Value

A Seurat object with rejoined layers and an `int_reduction`-named reduction ("cca", "rpca", or "harmony").

`integrate_seurat_object`

Integrate a merged Seurat object across batches.

Description

The `integrate` step of the pipeline in one call. It plays the same role for a merged object that `create_seurat_object()` plays for a single sample.

Usage

```
integrate_seurat_object(
  x,
  assay = "RNA",
  int_reduction = "cca",
  num_dim,
  batch_var = "orig.ident",
  num_variable_genes = 3000,
  num_neighbors = 30,
  k_anchor = 10,
  k_weight = 100,
  log_file = NULL,
  ...
)
```

Arguments

<code>x</code>	Merged Seurat object, such as the output of <code>merge_seurat_objects()</code> .
<code>assay</code>	Assay to integrate. Passed straight through to <code>integrate_layers()</code> .
<code>int_reduction</code>	Integration method: "cca", "rpca", or "harmony".
<code>num_dim</code>	Number of dimensions to use (5-50).
<code>batch_var</code>	Metadata column identifying the batch.
<code>num_variable_genes</code>	Number of variable features per batch.
<code>num_neighbors</code>	Neighbors for UMAP.
<code>k_anchor</code>	Neighbors to use when picking anchors. Passed straight through to <code>integrate_layers()</code> - see there for defaults/caveats (ignored by "harmony").
<code>k_weight</code>	Neighbors to use when weighting the corrections. Passed straight through to <code>integrate_layers()</code> - see there for defaults/caveats (also ignored by "harmony").
<code>log_file</code>	Filename for the log file.
<code>...</code>	Passed straight through to <code>integrate_layers()</code> - extra method-specific tuning arguments (harmony's <code>theta/lambda/sigma/...</code> , or <code>cca/rpca's k.filter/sample.tree/...</code>).

Details

The function runs these steps:

1. Split the assay layers by batch and prepare them (`split_layers_by_batch()`).
2. Report how much the per-batch variable genes overlap (`plot_var_genes_euler()`, `plot_var_genes_upset()`).
3. Correct the batch effect (`integrate_layers()`).
4. Compute a first-pass tSNE/UMAP on the corrected embedding.

The tSNE/UMAP from step 4 is a preview, capped at `num_dim`. It differs from the real one `cluster_seurat_object()` computes later over its own `num_dim`. `cluster_seurat_object()` has no default reduction. Pass it this function's `int_reduction` name explicitly. Otherwise it may use the uncorrected "pca" reduction that `split_layers_by_batch()` also leaves on the object.

Value

A Seurat object with rejoined layers, an `int_reduction`-named reduction ("cca", "rpca", or "harmony"), and "tsne"/"umap" reductions computed from it. Also writes the variable-gene overlap, QC distribution, integrated-embedding, and tSNE/UMAP plots to the working directory.

merge_metadata	<i>Function to merge two metadata tables together.</i>
----------------	--

Description

x and y must not share column names other than cell - stop()s if they do, rather than letting left_join() silently rename the collision to <col>.x/<col>.y.

Usage

```
merge_metadata(x, y, log_file = NULL)
```

Arguments

x	A Seurat object or tibble with cell IDs (a cell column, or as rownames).
y	A tibble with cell IDs (a cell column, or as rownames).
log_file	A log filename.

Value

A metadata file merged on cell identifiers.

merge_seurat_objects	<i>Merge multiple Seurat objects into one.</i>
----------------------	--

Description

Merges a list of Seurat objects, rejoins the split layers that Seurat 5 leaves behind, and removes genes that are detected in too few cells. Features of the non-RNA assays (such as ADT and HTO) are always kept.

Usage

```
merge_seurat_objects(seurat_object_list, min_cells = NULL, log_file = NULL)
```

Arguments

seurat_object_list	List of Seurat objects and/or analysis directory paths (mixing both is fine).
min_cells	Keep genes detected in at least this many cells. If NULL, 10 is used, or 0.1% of the cells when there are more than 50,000 of them.
log_file	Filename for the log file.

Details

Elements of `seurat_object_list` may be Seurat objects, or paths to an analysis directory containing a `seurat_obj.qs2` (as written by the CLI). Which one it is does not matter to the caller: this function resolves each element to an object first.

Once every input is an object, this function strips each one's own reductions, scaled data, and clustering columns before merging. A per-sample PCA, tSNE/UMAP, or clustering — written by `create_seurat_object()/cluster_seurat_object()` — is meaningless once merged. `normalize_counts()/run_pca()` recompute all of it downstream anyway.

Writes `metrics-distribution.png`, a per-sample QC violin plot, to the working directory.

Value

A merged Seurat object.

<code>normalize_counts</code>	<i>Normalize the counts, select variable features, and scale.</i>
-------------------------------	---

Description

A thin wrapper that picks the right Seurat function for method and fixes the parameters this pipeline uses, so either path hands back a fully processed object — no separate variance step needed. `method = "sct"` already worked this way natively: `SCTransform()` produces the counts, variable features and scaled data in one pass. `method = "log"` now mirrors that explicitly: `NormalizeData()` followed by `FindVariableFeatures()` and `ScaleData()`.

Usage

```
normalize_counts(
  x,
  method = "log",
  num_variable_genes = 3000,
  assay = "RNA",
  log_file = NULL
)
```

Arguments

<code>x</code>	A Seurat object.
<code>method</code>	"log" for <code>NormalizeData() + FindVariableFeatures() + ScaleData()</code> (log-normalized to a scale factor of 10,000), or "sct" for <code>SCTransform()</code> .
<code>num_variable_genes</code>	Number of variable features.
<code>assay</code>	Assay to normalize.
<code>log_file</code>	Log file.

Details

No covariates are regressed out (`vars.to.regress = NULL`) for either path. Germain et al. (2020) found that the "common practice of regressing out cell covariates, such as the detection rate or proportion of mitochondrial reads[,] nearly always had a negative impact."

On the log path, this function drops any `vf_*` columns already on assay's feature meta data before recomputing them. An earlier call can leave these columns behind: each sample's own pre-merge run, or a merge step's preview run. Left alone, they would just accumulate call after call. See [variable_features_by_batch\(\)](#) for what reading an accumulated column list, instead of a current one, leads to.

`SCTransform()` never writes this kind of per-gene bookkeeping column. So the sct path has nothing to clean up. The check runs the same way for both paths; it is a no-op for sct, not something conditioned on method.

Value

The processed Seurat object. Also writes the variable-feature plot (`variance-features.png`) to the working directory.

`plot_cluster_markers_heatmap`

Heatmap of the top cluster markers, at several top-N cutoffs.

Description

Heatmap of the top cluster markers, at several top-N cutoffs.

Usage

```
plot_cluster_markers_heatmap(x, markers_tbl, num_genes, filename_base)
```

Arguments

<code>x</code>	Seurat object with an identity set.
<code>markers_tbl</code>	Marker table, such as from calculate_cluster_markers() - either the standard <code>log2FC/cluster</code> columns, or the pairwise <code>log2FC_min/cluster</code> columns.
<code>num_genes</code>	Top-N genes per cluster to include, one heatmap per value.
<code>filename_base</code>	Path prefix for the output files.

Value

Nothing; called for its file output.

plot_cluster_markers_top

Plot a gene list in multiple ways: UMAP, dot plot, violin, and per-cluster bar plot.

Description

Plot a gene list in multiple ways: UMAP, dot plot, violin, and per-cluster bar plot.

Usage

```
plot_cluster_markers_top(
  x,
  genes,
  filename_base,
  color_scheme = get_color_scheme("clusters")
)
```

Arguments

x	Seurat object with an identity set.
genes	Genes to plot.
filename_base	Path prefix for the output files.
color_scheme	(optional) Named vector of cluster colors, used by the violin and bar plots. Defaults to <code>get_color_scheme("clusters")</code> .

Value

Nothing; called for its file output.

plot_clusters

Plot a cluster resolution on tSNE and UMAP.

Description

A single cluster says nothing, and past the color scheme's length the colors start repeating, so nothing is plotted outside `2:length(color_scheme)` clusters.

Usage

```
plot_clusters(
  x,
  resolution,
  filename_base,
  color_scheme = get_color_scheme("clusters")
)
```

Arguments

x	Seurat object.
resolution	Metadata column (or bare resolution value, resolved via <code>check_identity_column()</code>) to set as the identity before plotting.
filename_base	Path prefix for the two plots (-tsne/-umap are appended by <code>plot_dr_group()</code>).
color_scheme	(optional) Named vector of cluster colors. Defaults to <code>get_color_scheme("clusters")</code> .

Value

x with resolution set as its identity.

plot_distribution	<i>Plot the distribution of specified features/variables.</i>
-------------------	---

Description

Plot the distribution of specified features/variables.

Usage

```
plot_distribution(x, features, grouping, color_scheme = NULL)
```

Arguments

x	Seurat object or metadata.
features	Vector of features to plot (such as "nGene", "nUMI", "percent.mito").
grouping	X.
color_scheme	(optional) Vector of colors.

Value

A vector of colors.

plot_dr_feature	<i>Plot a single feature overlaid on a dimensionality reduction.</i>
-----------------	--

Description

One point per cell, colored by expression/abundance of feature (a gene, antibody, or any other value `FeaturePlot()` can fetch). Unlike passing a multi-color ramp straight to `FeaturePlot(cols = )`, the ramp is applied afterward with `scale_color_gradientn()` so the color scale stays continuous - see the "cols" note below.

Usage

```
plot_dr_feature(
  x,
  feature,
  assay = NULL,
  reduction = "umap",
  cells = NULL,
  color_scheme = NULL,
  pt_size = NULL
)
```

Arguments

<code>x</code>	Seurat object.
<code>feature</code>	Feature name (gene, antibody, or metadata column) to plot.
<code>assay</code>	(optional) Assay to fetch feature from. <code>NULL</code> uses <code>DefaultAssay(x)</code> .
<code>reduction</code>	Dimensionality reduction to plot on (e.g. "umap", "tsne", "pca").
<code>cells</code>	(optional) Cells to plot, in the given order (controls z-order of overlapping points). Defaults to a random shuffle of all cells so no single cell/group dominates due to plot order.
<code>color_scheme</code>	(optional) Vector of colors for the low-to-high expression gradient.
<code>pt_size</code>	(optional) Point size. Defaults to <code>get_dr_point_size(x)</code> .

Value

A ggplot object.

Seurat's `FeaturePlot(cols = )` quirk

As of Seurat 5, passing a `cols` vector of anything other than exactly 2 colors makes `FeaturePlot()` bin the continuous values into 2 groups before coloring (`cut(x, breaks = 2)`), which turns a smooth expression gradient into a flat low/high split. Seurat 4 only did this when `cols` had exactly 2 colors, so a long custom ramp used to pass through untouched. This function sidesteps the quirk (and the version difference) entirely by never passing a multi-color `cols` to `FeaturePlot()` - the ramp is layered on afterward instead.

plot_dr_group

Scatter plot of a reduction, colored by a grouping variable.

Description

The categorical counterpart of `plot_dr_feature()`: cells are colored by a metadata column (or by the current identities) rather than by a continuous feature. Cells are shuffled so no one group is drawn on top of the others.

Usage

```
plot_dr_group(
  x,
  reduction = "umap",
  group_by = NULL,
  color_scheme = NULL,
  pt_size = NULL,
  na_value = "grey50",
  file_prefix = NULL,
  file_format = c("png", "pdf"),
  width = NULL,
  height = 6
)
```

Arguments

x	Seurat object.
reduction	Dimensionality reduction to plot on (e.g. "umap", "tsne", "pca").
group_by	(optional) Metadata column to color by. NULL uses the current identities.
color_scheme	(optional) Vector of colors. Defaults to the "clusters" scheme.
pt_size	(optional) Point size. Defaults to <code>get_dr_point_size(x)</code> .
na_value	Color for cells with no value in group_by.
file_prefix	(optional) Path to save to.
file_format	File formats to write when file_prefix is given.
width	(optional) Saved width in inches. NULL measures the plot's own legend and widens to fit it - see <code>save_dr_plot()</code> .
height	Saved height in inches.

Value

A ggplot object.

plot_dr_umap_clusters *Plot a UMAP colored by one or more cluster/grouping variables.*

Description

Plot a UMAP colored by one or more cluster/grouping variables.

Usage

```
plot_dr_umap_clusters(
  x,
  cluster_var,
  color_scheme = get_color_scheme("clusters")
)
```

Arguments

x	Seurat object with a "umap" reduction.
cluster_var	Metadata column(s) to color by, as a single string (comma-separated for more than one).
color_scheme	(optional) Named vector of colors. Defaults to get_color_scheme("clusters").

Value

Nothing; called for its file output, one dr-umap-<cluster_var>.png per variable.

plot_metrics_correlations
Scatter plots of the per-cell QC metrics against each other.

Description

One panel per pair in pairs, colored by sample. No legend, since this only ever runs on a single sample - see [filter_cells\(\)](#). Pairs are a curated list rather than every combination of a metric vector, since that would make the panel count quadratic in the number of metrics (5 metrics -> 10 panels via `combn()`) and most pairs are not biologically meaningful together.

Usage

```
plot_metrics_correlations(
  x,
  pairs = list(c("total_counts", "detected_genes"), c("pct_mito", "detected_genes"),
    c("pct_mito", "pct_ribo")),
  group_by = "orig.ident",
  color_scheme = NULL,
```

```

    file = NULL,
    width = 16,
    height = 5
  )

```

Arguments

<code>x</code>	Seurat object.
<code>pairs</code>	A list of length-2 character vectors, <code>c(feature1, feature2)</code> , one per panel.
<code>group_by</code>	Metadata column to color the points by.
<code>color_scheme</code>	(optional) Named vector of colors. Defaults to the "samples" scheme, named by the sorted levels of <code>group_by</code> .
<code>file</code>	Path to save the plot to. NULL returns it without writing. Parent directories are created as needed.
<code>width, height</code>	Saved size in inches. The default is wide and short: one square panel per pair of metrics, side by side.

Value

A combined plot, one panel per pair.

plot_metrics_distribution

Violin plots of the per-cell QC metrics.

Description

One violin panel per metric, grouped by sample. The metrics are metadata columns, so `layer` is set explicitly: `VlnPlot()` otherwise searches for the "data" layer, which does not exist before `NormalizeData()` has been run.

Usage

```

plot_metrics_distribution(
  x,
  metrics,
  group_by = "orig.ident",
  color_scheme = NULL,
  file = NULL,
  width = 12,
  height = 6
)

```

Arguments

x	Seurat object.
metrics	Metadata columns to plot, one panel each. No default - the caller decides, since the right set depends on context (e.g. a single sample can afford pct_ribo/pct_hb alongside detected_genes/total_counts/pct_mito; a multi-sample plot usually can't). A pct_* metric that is under 1% in every cell is skipped (a flat near-zero violin says nothing) - its range is reported with a message instead.
group_by	Metadata column to group the violins by.
color_scheme	(optional) Named vector of colors. Defaults to the "samples" scheme, named by the sorted levels of group_by.
file	Path to save the plot to. NULL returns it without writing. Parent directories are created as needed.
width, height	Saved size in inches. Widen it when there are many samples.

Value

A combined plot, one panel per plotted metric.

plot_var_genes_euler *Euler diagram of the variable genes shared between batches.*

Description

Becomes unreadable and can take a very long time to fit for many sets, so returns NULL when there are too many batches.

Usage

```
plot_var_genes_euler(var_genes_list, color_scheme = NULL, max_sets = 8)
```

Arguments

var_genes_list	Named list of variable gene vectors, one element per batch.
color_scheme	(optional) Vector of colors.
max_sets	Return NULL when there are more sets than this.

Value

A plot, or NULL if there are too many sets to draw.

`plot_var_genes_upset` *UpSet plot of the variable genes shared between batches.*

Description

UpSet plot of the variable genes shared between batches.

Usage

```
plot_var_genes_upset(var_genes_list, nsets = 50, nintersects = 15)
```

Arguments

`var_genes_list` Named list of variable gene vectors, one element per batch.
`nsets` Maximum number of sets to include.
`nintersects` Maximum number of intersections to show.

Value

An UpSet plot.

`profile_object_size` *Report the size of an object's slots, largest first.*

Description

Works on any S4 object or list, not just Seurat objects. It descends into S4 slots and list elements, bounded by `max_depth`. So a Seurat object, an assay pulled from one (`x[["RNA"]]`), a `DimReduc`, or a plain nested list all get the same recursive breakdown.

Usage

```
profile_object_size(x, max_depth = 3)
```

Arguments

`x` Any object - a Seurat object, a piece pulled from one (an assay via `x[["RNA"]]`, a reduction via `x[["pca"]]`, ...), a plain list, or anything else `object.size()` accepts.
`max_depth` How many levels to descend before reporting a subtree's total size instead of continuing to break it down.

Details

An Assay5's layers slot is itself a list. So `layers > counts/layers > data/layers > scale.data` are sized separately, not lumped into one layers total.

A matrix, data frame, or atomic vector is neither S4 nor a list. Passing one of these just reports its own size, the same as calling `object.size()` directly. There is nothing to descend into.

Value

A tibble with one row per slot/element, largest first: path (its position, " > "-separated) and size (human-readable, e.g. "1.2 Mb").

read_counts_file	<i>Read in Gene Expression and Antibody Capture data from a 10x Genomics Cell Ranger sparse matrix or from a text file.</i>
------------------	---

Description

Read in Gene Expression and Antibody Capture data from a 10x Genomics Cell Ranger sparse matrix or from a text file.

Usage

```
read_counts_file(path, sample_name, log_file = NULL)
```

Arguments

path	Path to directory containing 10x matrix, or path to a text file.
sample_name	A character that will be used as a prefix for all cell names.
log_file	Filename for the log file.

Value

Named list of matrices. One matrix for each data type. Currently the only two data types are 'Gene Expression' and 'Antibody Capture'

resolve_seurat_object	<i>Resolve a Seurat object, given either the object itself or a path to one on disk.</i>
-----------------------	--

Description

Meant for parameters that accept either an in-memory Seurat object or a serialized one - resolving here means callers do not need to check which they got themselves.

Usage

```
resolve_seurat_object(x)
```

Arguments

x	A Seurat object, a path to one saved as .rds or .qs2, or a directory containing exactly one such file.
---	--

Value

A Seurat object.

run_dr	<i>Run dimensionality reduction: pca, tsne, or umap</i>
--------	---

Description

Run dimensionality reduction: pca, tsne, or umap

Usage

```
run_dr(x, dr_method, suffix = "", assay = "RNA", ...)
```

Arguments

x	A matrix or a Seurat object.
dr_method	Dimensionality reduction method ("pca", "tsne", or "umap").
suffix	Tag inserted into the reduction name and key.
assay	Assay to use (Seurat methods). Always passed explicitly here rather than left NULL, which would otherwise fall back on whatever DefaultAssay(x) happens to be.
...	Passed to the chosen reduction function.

Details

run_dr() is a thin dispatcher: it routes to run_pca(), run_tsne(), or run_umap() based on dr_method, forwarding suffix and anything in It only carries the parameters common to every method; method-specific ones (num_pcs, num_neighbors, reduction, ...) are passed through ... to the chosen function. Each of those functions is itself generic: a plain matrix takes the .default method (irlba/Rtsne/uwot), a Seurat object takes the .Seurat method (native RunPCA()/RunTSNE()/RunUMAP()), storing a "<method><suffix>" reduction).

Value

The reduced data (matrix input) or a Seurat object with the reduction added. For a Seurat object, also writes the diagnostic plot(s) for the chosen method to the working directory: PCA gets a sample-colored scatter and an elbow plot (plus a heatmap past 15 components); tSNE/UMAP get a sample-colored scatter.

See Also

run_pca(), run_tsne(), run_umap()

run_pca

Run PCA

Description

The default method runs irlba::prcomp_irlba() on a matrix; the Seurat method runs Seurat::RunPCA() on the object and stores a "pca<suffix>" reduction.

Usage

```
run_pca(x, ...)

## Default S3 method:
run_pca(x, num_pcs = 50, suffix = "", ...)

## S3 method for class 'Seurat'
run_pca(
  x,
  num_pcs = 50,
  suffix = "",
  assay = "RNA",
  features = NULL,
  var_features = FALSE,
  ...
)
```

Arguments

<code>x</code>	A matrix (default) or a Seurat object.
<code>...</code>	Passed to the underlying function.
<code>num_pcs</code>	Number of principal components.
<code>suffix</code>	Tag inserted into the reduction/column names.
<code>assay</code>	Assay to use (Seurat method). Always set explicitly.
<code>features</code>	Explicit features to use (Seurat method).
<code>var_features</code>	Use the assay's variable features (Seurat method).

Value

A named list (default) or a Seurat object with the reduction added (Seurat method). The Seurat method also writes the diagnostic plots to the working directory: a sample-colored scatter and an elbow plot, plus a heatmap past 15 components.

run_tsne

Run TSNE

Description

The default method runs `Rtsne::Rtsne()` on a matrix; the Seurat method runs `Seurat::RunTSNE()` on the object and stores a "tsne<suffix>" reduction.

Usage

```
run_tsne(x, ...)

## Default S3 method:
run_tsne(x, seed.use = 1, dim.embed = 2, suffix = "", ...)

## S3 method for class 'Seurat'
run_tsne(
  x,
  seed.use = 1,
  dim.embed = 2,
  suffix = "",
  assay = "RNA",
  reduction = NULL,
  num_dim = NULL,
  features = NULL,
  var_features = FALSE,
  file_format = c("png", "pdf"),
  ...
)
```

Arguments

<code>x</code>	A matrix (default) or a Seurat object.
<code>...</code>	Passed to the underlying function.
<code>seed.use</code>	Seed to use.
<code>dim.embed</code>	Number of tSNE embeddings to return.
<code>suffix</code>	Tag inserted into the reduction/column names.
<code>assay</code>	Assay to use (Seurat method). Always set explicitly.
<code>reduction</code>	Existing reduction to take dimensions from (Seurat method).
<code>num_dim</code>	Number of dimensions to take from reduction (Seurat method).
<code>features</code>	Explicit features to use (Seurat method).
<code>var_features</code>	Use the assay's variable features (Seurat method).
<code>file_format</code>	File formats to write the scatter plot in (Seurat method).

Value

A tSNE embedding matrix (default) or a Seurat object with the reduction added. The Seurat method also writes a sample-colored scatter of the result to the working directory, as `dr.<reduction>.png/.pdf` (or just `.png` if `file_format = "png"`).

run_umap

Run UMAP

Description

The default method runs `uwot::umap2()` on a matrix; the Seurat method runs `Seurat::RunUMAP()` with `umap.method = "uwot2"` (also `uwot::umap2()`) and stores a `"umap<suffix>"` reduction. Defaults match `Seurat::RunUMAP()` (30 neighbors, minimum distance 0.3) rather than the lower `uwot` defaults.

Usage

```
run_umap(x, ...)

## Default S3 method:
run_umap(x, num_neighbors = 30, min_dist = 0.3, suffix = "", ...)

## S3 method for class 'Seurat'
run_umap(
  x,
  num_neighbors = 30,
  min_dist = 0.3,
  suffix = "",
  assay = "RNA",
  reduction = NULL,
```

```

    num_dim = NULL,
    features = NULL,
    var_features = FALSE,
    graph = NULL,
    file_format = c("png", "pdf"),
    ...
)

```

Arguments

<code>x</code>	A matrix (default) or a Seurat object.
<code>...</code>	Passed to the underlying function.
<code>num_neighbors</code>	Number of neighbors.
<code>min_dist</code>	Minimum distance between points in the embedding.
<code>suffix</code>	Tag inserted into the reduction/column names.
<code>assay</code>	Assay to use (Seurat method). Always set explicitly.
<code>reduction</code>	Existing reduction to take dimensions from (Seurat method).
<code>num_dim</code>	Number of dimensions to take from reduction (Seurat method).
<code>features</code>	Explicit features to use (Seurat method).
<code>var_features</code>	Use the assay's variable features (Seurat method).
<code>graph</code>	Graph to use as input (Seurat method).
<code>file_format</code>	File formats to write the scatter plot in (Seurat method).

Value

A UMAP embedding matrix (default) or a Seurat object with the reduction added. The Seurat method also writes a sample-colored scatter of the result to the working directory, as `dr.<reduction>.png/.pdf` (or just `.png` if `file_format = "png"`).

save_counts	<i>Save the counts matrix as a single table.</i>
-------------	--

Description

Layers left split by a `merge()` are rejoined first, since `LayerData()` would otherwise return only the first of them, and warn rather than error while doing it.

Usage

```
save_counts(x, assay = "RNA", layer = "data", digits = 3, file)
```

Arguments

x	A Seurat object.
assay	Assay to retrieve the counts from.
layer	Layer to retrieve. The default "data" layer holds the normalized and log-transformed expression used for visualization and most differential expression tests.
digits	Number of digits to round the values to. Ignored for a layer that already holds whole numbers, such as raw counts.
file	Path to write the table to as a csv, or NULL to skip writing and just return the tibble. Has no default: every caller states where the table goes.

Value

A tibble with one row per gene.

save_dr_plot	<i>Save a reduction plot, widened to fit its own legend.</i>
--------------	--

Description

Unlike `dr_plot_width()` (a label-count guess made before the plot exists), this measures the legend actually rendered on plot - so long label text widens the saved image even when the label count is small. Requires an open graphics device to get accurate text metrics, so it opens and closes a throwaway one around the measurement.

Usage

```
save_dr_plot(
  plot,
  file_prefix,
  file_format = c("png", "pdf"),
  width = NULL,
  height = 6,
  panel_width = height
)
```

Arguments

plot	A ggplot object, such as from <code>plot_dr_group()</code> . Forwarded straight to <code>ggsave(plot =)</code> .
file_prefix	Path to save to without an extension.
file_format	File formats to write.
width	(optional) Saved width in inches. NULL measures plot's legend and adds it to panel_width.
height	Saved height in inches.
panel_width	Width of the plot panel itself, excluding the legend, in inches. Only used when width is NULL.

Value

plot, invisibly.

save_metadata	<i>Save the cell metadata and the reduction embeddings as a single table.</i>
---------------	---

Description

Joins the requested reductions onto the cell metadata on a cell column. Reductions that are not present in the object are skipped, so this works before and after the dimensionality reduction steps have run.

Usage

```
save_metadata(
  x,
  reduction = c("tsne", "umap"),
  digits = 3,
  file = "metadata.csv.gz"
)
```

Arguments

x	A Seurat object.
reduction	Reductions to join onto the metadata.
digits	Number of digits to round the embeddings to.
file	Path to write the table to as a csv. Set to NULL to skip writing and just return the tibble.

Value

A tibble with one row per cell.

set_identity	<i>Set identity of the Seurat object.</i>
--------------	---

Description

Wrapper for SeuratObject::Idents() with extra safety checks.

Usage

```
set_identity(x, identity_column)
```

Arguments

`x` A Seurat object.

`identity_column` The name of the identity column to pull from object metadata.

Value

A Seurat object with an updated identity set.

`split_layers_by_batch` *Split a Seurat object into per-batch layers and prepare it for integration.*

Description

Drops the scaled data, reductions, graphs, and clustering columns, then splits the assay layers by a metadata column — splitting the layers rather than the object is what the Seurat 5 `IntegrateLayers()` workflow expects.

Usage

```
split_layers_by_batch(
  x,
  batch_var = "orig.ident",
  assay = "RNA",
  num_variable_genes = 3000,
  num_dim = 50,
  log_file = NULL
)
```

Arguments

`x` Seurat object.

`batch_var` Metadata column to split the layers by.

`assay` Assay to split.

`num_variable_genes` Number of variable features. Matches the single-sample default.

`num_dim` Number of principal components to compute.

`log_file` Filename for the log file.

Details

Everything after the split runs through the same verbs as the single-sample path (`normalize_counts()`, `run_pca()`), so the two workflows cannot drift apart on parameters. `integrate_layers()` then consumes the resulting "pca" reduction.

Value

A Seurat object with split layers and a "pca" reduction.

transfer_labels	<i>Transfer labels from a reference Seurat object</i>
-----------------	---

Description

Reference-based annotation wrapping Seurat's reference-mapping workflow ([Seurat::FindTransferAnchors\(\)](#) + [Seurat::TransferData\(\)](#)) to annotate query with labels (cell type, or any other categorical metadata) from an already integrated and annotated reference object. Optionally also projects query onto the reference's UMAP via [Seurat::MapQuery\(\)](#).

Usage

```
transfer_labels(
  query,
  ref,
  query_label_col,
  ref_label_col,
  query_assay,
  ref_assay,
  normalization,
  ref_reduction = "pca",
  num_dim,
  anchor_reduction = "pcaproject",
  ref_umap = FALSE,
  verbose = TRUE
)
```

Arguments

query	Query Seurat object.
ref	Reference Seurat object - already integrated and annotated.
query_label_col	Metadata column name added to query for the transferred labels.
ref_label_col	Metadata column in ref holding the labels to transfer (cell type, or any other category).
query_assay	Assay in query to use for anchor finding. "RNA" (log-normalized) or "SCT", depending on how query was built. No default, like the params below: this and ref_assay/normalization/ num_dim all describe how a specific object was built.
ref_assay	Assay in ref to use for anchor finding.
normalization	Normalization method shared by ref and query ("log"/"LogNormalize" or "sct"/"SCT"). Seurat::FindTransferAnchors() takes one value for both objects, so ref and query must actually share a method.

ref_reduction	Reduction in ref to project query onto.
num_dim	Number of dimensions to use for anchor finding and label transfer.
anchor_reduction	Dimensional reduction workflow <code>Seurat::FindTransferAnchors()</code> uses to find anchors.
ref_umap	If TRUE, also runs <code>Seurat::MapQuery()</code> and adds a "ref.umap" reduction to query, projecting it onto ref's UMAP. ref needs a UMAP built with <code>return.model = TRUE</code> , which this function does not check for - <code>MapQuery()</code> errors on its own if it is missing.
verbose	Print progress messages.

Value

query with `query_label_col` (a factor) added to its metadata, and a "ref.umap" reduction added if `ref_umap = TRUE`. Also writes the predictions table (`cell/predicted.id/prediction.score.max`) to `annotation/annotation-transfer-label-<query_label_col>.csv.gz`, and a `dr-<reduction>-<query_label_col>` UMAP colored by the transferred labels for each of "umap"/"ref.umap" already present on query.

variable_features_by_batch

Variable features of each batch of a layer-split object.

Description

After `split_layers_by_batch()` the per-batch variable features are stored as `vf_<method>_<layer>_variable` columns in the assay meta data rather than being reachable through `VariableFeatures()`, which returns the combined set.

Usage

```
variable_features_by_batch(x, assay = "RNA")
```

Arguments

x	Seurat object with split layers.
assay	Assay to read the variable features from.

Details

`normalize_counts()` now clears its own assay's `vf_*` columns before recomputing them. But an object built before that guard existed can still carry a whole history of them: each sample's own pre-merge run from `create_seurat_object()`, and the merge step's own post-merge preview run. `DietSeurat()` clears layers, reductions, and graphs, but never assay meta data. `merge()` disambiguates colliding names with a numeric suffix, for example `vf_vst_counts.1_variable`.

Matching every `_variable$` column would pick up that whole history along with the batches actually being asked about. This function instead anchors on `Layers(x[[assay]], search = "counts")`, the object's current split state. That keeps stale columns out, regardless of which generation of the object this is.

Value

A named list of variable gene vectors, one element per batch.

write_message	<i>Small function to write to message and to log file.</i>
---------------	--

Description

Small function to write to message and to log file.

Usage

```
write_message(message_str, log_file = NULL)
```

Arguments

message_str	A string to write as a message.
log_file	A log filename.

Value

A message and writes the message to the specified log file.

Examples

```
write_message(message_str = "Finished Step 1", log_file = "log.file.txt")
```

Index

`add_gene_class_percent`, 3
`add_gene_class_percent()`, 18
`add_seurat_assay`, 4
`add_seurat_assay()`, 11, 12
`as_data_frame_seurat`, 4

`calculate_cluster_expression`, 5
`calculate_cluster_markers`, 6
`calculate_cluster_markers()`, 24
`calculate_cluster_stats`, 6
`calculate_clusters`, 7
`calculate_clusters()`, 9
`check_identity_column`, 8
`check_identity_column()`, 26
`cluster_seurat_object`, 9
`cluster_seurat_object()`, 12, 21, 23
`create_color_vect`, 10
`create_seurat_object`, 10
`create_seurat_object()`, 9, 20, 23, 43

`differential_expression_per_cluster`, 12
`dr_plot_width`, 13
`dr_plot_width()`, 39

`filter_cells`, 14
`filter_cells()`, 11, 29

`geneset_score`, 15
`get_color_scheme`, 16
`get_dr_point_size`, 16
`get_test_counts_matrix`, 17

`import_mtx`, 17
`initialize_seurat_object`, 18
`initialize_seurat_object()`, 11, 14
`integrate_layers`, 19
`integrate_layers()`, 21, 41
`integrate_seurat_object`, 20
`integrate_seurat_object()`, 9, 19, 20

`merge_metadata`, 22
`merge_seurat_objects`, 22
`merge_seurat_objects()`, 21

`normalize_counts`, 23
`normalize_counts()`, 11, 12, 23, 41, 43

`plot_cluster_markers_heatmap`, 24
`plot_cluster_markers_top`, 25
`plot_clusters`, 25
`plot_distribution`, 26
`plot_dr_feature`, 27
`plot_dr_feature()`, 28
`plot_dr_group`, 28
`plot_dr_group()`, 13, 26, 39
`plot_dr_umap_clusters`, 29
`plot_metrics_correlations`, 29
`plot_metrics_correlations()`, 15, 18
`plot_metrics_distribution`, 30
`plot_metrics_distribution()`, 15, 18
`plot_var_genes_euler`, 31
`plot_var_genes_euler()`, 21
`plot_var_genes_upset`, 32
`plot_var_genes_upset()`, 21
`profile_object_size`, 32

`read_counts_file`, 33
`read_counts_file()`, 11
`resolve_seurat_object`, 34
`rlang::check_dots_used()`, 20
`run_dr`, 34
`run_pca`, 35
`run_pca()`, 12, 23, 41
`run_tsne`, 36
`run_tsne()`, 9, 10, 12
`run_umap`, 37
`run_umap()`, 9, 10, 12

`save_counts`, 38
`save_dr_plot`, 39

`save_dr_plot()`, [13](#), [28](#)
`save_metadata`, [40](#)
`save_metadata()`, [9](#), [12](#), [15](#)
`set_identity`, [40](#)
`set_identity()`, [5](#), [7](#)
`Seurat::FindAllMarkers()`, [6](#)
`Seurat::FindMarkers()`, [6](#)
`Seurat::FindTransferAnchors()`, [42](#), [43](#)
`Seurat::MapQuery()`, [42](#), [43](#)
`Seurat::TransferData()`, [42](#)
`split_layers_by_batch`, [41](#)
`split_layers_by_batch()`, [19–21](#), [43](#)

`transfer_labels`, [42](#)

`variable_features_by_batch`, [43](#)
`variable_features_by_batch()`, [24](#)

`write_message`, [44](#)