

Package: SpaMTP (via r-universe)

September 25, 2026

Type Package

Title Spatial Metabolomics and Multi-Omics Analysis

Version 0.99.10

Description Provides analysis, annotation, visualisation, and multi-omic integration methods for spatial metabolomics data. SpaMTP interoperates with Cardinal and SpatialExperiment containers, with optional Seurat conversion; supports indexed accurate-mass annotation using versioned RaMP resources; and implements spatial feature analysis, pathway enrichment, pathway-network visualisation, and alignment with spatial transcriptomics data.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Suggests BiocStyle, EnhancedVolcano, htmlTable, htmltools, kableExtra, knitr, patchwork, reticulate, rmarkdown, SeuratObject (>= 5.0.0), SpaMTPData (>= 0.99.5), testthat (>= 3.0.0), viridis, yaml

Config/testthat/edition 3

biocViews MassSpectrometry, ImagingMassSpectrometry, Metabolomics, Spatial, Transcriptomics, Alignment

Imports ape, BiocSingular, bluster, Cardinal, cowplot, data.table, digest, dplyr, DropletUtils, edgeR, fgsea, FNN, future, future.apply, gg dendro, ggplot2, ggridges, grDevices, graphics, grid, httr, htmlwidgets, jsonlite, limma, magick, magrittr, Matrix, mclust, methods, MSnbase, natural sort, parallel, pheatmap, plotly, png, purrr, rgsolin, rlang, RSpectra, scales, scater, scan, sf, shiny, SingleCellExperiment, S4Vectors, SpaMTPdb (>= 0.99.4), SpatialExperiment, stats, stringr, SummarizedExperiment, tibble, tidyr, tools, utils, withr

URL <https://github.com/SpaMTP-project/SpaMTP>,
<https://github.com/GenomicsMachineLearning/SpaMTP>,

<https://genomicsmachinelearning.github.io/SpaMTP/>,
<https://doi.org/10.1038/s41592-026-03140-8>

BugReports <https://github.com/SpaMTP-project/SpaMTP/issues>

Depends R (>= 4.5.0)

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libglpk-dev make libmagick++-dev gsfonts libicu-dev libpng-dev libuv1-dev libxml2-dev libnetcdf-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev zlib1g-dev

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

Date/Publication 2026-09-25 05:15:39 UTC

RemoteUrl <https://github.com/BiocStaging/SpaMTP>

RemoteRef HEAD

RemoteSha 2c54b7bdefaec3fd714d185655bedeac56ba133a

Contents

add_backslashes_to_specialfeatures	5
addCustomMZAnnotations	5
addFMP10Annotations	6
addSMImage	8
addSpatialImage	9
addSSCAnnotation	10
addTranscriptome	11
adduct_file	12
adductRules	12
alignSpatialOmics	13
analyzeSpaMTPRegions	14
annotateBigData	15
annotateGeneIdentifiers	17
annotateMZ	18
annotateSM	19
annotateSMILESStructure	21
annotateTable	22
annotationInfo	24
applySpatialAlignment	24
asCardinal	28
asSpatialExperiment	29
binMetabolites	30
binnedCardinalToSeurat	30
binSpaMTP	31
buildGeneMappingIndex	33
buildMZAnnotationIndex	34
buildPathwayIndex	35

calculate_bounds	37
calculateAnnotationStatistics	38
calculateSingleAnnotationStatistics	40
cardinalToSeurat	41
cardinalToSpatialExperiment	42
check_and_truncate_adduct_vector	43
check_cardinal_version	44
checkAlignment	44
combineTransformations	45
convertSeuratToCardinal	46
createMergedModalityAssay	47
createPathwayAssay	48
createPathwayObject	51
db_adduct_filter	53
deconvolveSMILES	54
demsHeatmap	55
densityMap	57
experimentAccess	58
findAllDEMs	59
findCorrelatedFeatures	61
findDuplicateAnnotations	63
findNearestMZ	63
findRegionalPathways	64
findSpatiallyVariableMetabolites	67
fishersPathwayAnalysis	68
formula_filter	71
get_analytes_db	71
getKmeanClusters	72
getMetaspace	73
getMZMetadata	74
getRefinedAnnotations	75
getSpatiallyVariableMetabolites	75
getSquareCoordinates	76
imageMZAnnotationPlot	77
imageMZPlot	79
interactiveSpatialPlot	81
kNeighborsGraph	82
labels_to_show	83
loadMetaspace	83
loadSM	84
loadSpaMTPDatabase	85
malDiMatrixProfiles	87
malDiMatrixRules	87
mapGeneIdentifiers	88
mapSpatialOmics	90
massIntensityPlot	91
metaspaceClient	93
metaspaceToFeatureMatrix	93

multiOmicIntegration	94
mzBoxPlot	95
mzRidgePlot	97
mzViolinPlot	98
normalizeSMDData	99
pathwayNetworkPlots	101
pixelPlot	105
plot3DFeature	105
plotPathways	107
plotPathwaysSpatially	109
plotRegionalPathways	111
plotSinglePathway	112
plotSinglePathwaySpatially	113
plotSpatialFeature	115
ppm_error	117
ppm_range_match	117
predictAdductsFromSMILES	118
print.spamtp_mz_index	119
Pseudo_msms	119
queryMZAnnotationIndex	120
reaction_type	121
readSMMatrix	122
refineLipids	123
renderSpaMTPReport	124
runDE	125
runMetabolicPCA	126
runPooling	128
runRAMPGeseca	129
runSpaMTPWorkflow	131
runSpatialGraphPCA	133
savePheatmapAsPDF	135
saveSpaMTPData	135
scaleSMDData	137
searchAnnotations	138
selectROIs	139
seuratToSingleCellExperiment	140
seuratToSpatialExperiment	141
spaMTPDatabaseInfo	142
spatialExperimentToCardinal	142
spatialExperimentToSeurat	143
spatialMZAnnotationPlot	144
spatialMZPlot	145
spectralBinning	147
statPlot	148
subsetMZFeatures	149
subsetSPM	149
tmmNormalize	150
visualisePathways	151

Index**153**

`add_backslashes_to_specialfeatures`*Adds in backslashes required to take into account special using grepl such as brackets and +*

Description

Adds in backslashes required to take into account special using grepl such as brackets and +

Usage

```
add_backslashes_to_specialfeatures(input_string)
```

Arguments

`input_string` Character string requiring additional backslashes

Value

Character string containing double backslashes around special features

`addCustomMZAnnotations`*Assign custom annotations to m/z values*

Description

Adds custom metabolite annotations to respective m/z values (ideal for specific matrices such as FMP10).

Usage

```
addCustomMZAnnotations(  
  data,  
  annotations,  
  assay = "main",  
  return.only.annotated = FALSE,  
  mass.threshold = 0.05,  
  annotation.column = "all_IsomerNames"  
)
```

Arguments

<code>data</code>	Bioconductor experiment containing m/z intensity values.
<code>annotations</code>	data.frame containing two columns named 'annotation' and 'mass'. These columns should contain the custom metabolite annotation and the relative m/z mass respectively.
<code>assay</code>	Primary or alternative experiment whose <code>rowData()</code> receives the annotations (default = "main").
<code>return.only.annotated</code>	Boolean defining whether to return a Bioconductor experiment containing only successfully annotated m/z values (default = FALSE).
<code>mass.threshold</code>	Numeric value defining the acceptable threshold (plus-minus) between the custom annotations and the actual m/z values contained within the SpaMTP object (default = 0.05).
<code>annotation.column</code>	Name of the <code>rowData()</code> column receiving annotations (default = "all_IsomerNames").

Value

Bioconductor experiment containing the custom annotations stored in the feature metadata dataframe.

Examples

```
utils::str(formals(addCustomMZAnnotations))
# annotated_data <- addCustomMZAnnotations(SpaMTP.obj, annotation.df)
```

`addFMP10Annotations` *Annotates FMP10 matrix data*

Description

Adds metabolite annotations to respective m/z values generated using an FMP10 matrix. This is done based on a curated FMP10 matrix database.

Usage

```
addFMP10Annotations(
  obj,
  only.fmp.adduct = FALSE,
  add.custom.annotation = NULL,
  assay = "main",
  return.only.annotated = FALSE,
  mass.threshold = 0.05,
  annotation.column = "all_IsomerNames",
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL
)
```

Arguments

<code>obj</code>	Bioconductor experiment containing m/z intensity values for annotation. Data should be generated with a FMP10 matrix.
<code>only.fmp.adduct</code>	Boolean indicating if only metabolites with FMP10+ adducts (+FMP10, +2FMP10, etc.) should be assigned to m/z values (default = FALSE).
<code>add.custom.annotation</code>	data.frame containing addition FMP10 metabolite annotations that are not in the current FMP10 database. Note: this data.frame must contain these columns ("mass", "annotation", "Adduct", "Formula", "Isomers", "Isomers_IDs"). If set to NULL, only reference FMP10 database will be used (default = NULL).
<code>assay</code>	Primary or alternative experiment whose <code>rowData()</code> receives the annotations (default = "main").
<code>return.only.annotated</code>	Boolean defining whether to return a Bioconductor experiment containing only successfully annotated m/z values (default = FALSE).
<code>mass.threshold</code>	Numeric value defining the acceptable threshold (plus-minus) between the custom annotations and the actual m/z values contained within the SpaMTP object (default = 0.05).
<code>annotation.column</code>	Name of the <code>rowData()</code> column receiving annotations (default = "all_IsomerNames").
<code>database</code>	Optional named list of database resources, normally created by loadSpaMTPDatabase() .
<code>database_version</code>	SpaMTPdb/RaMP version used for annotation lookup.
<code>database_source</code>	Database source; see loadSpaMTPDatabase() .
<code>database_local_dir</code>	Optional staged SpaMTPdb resource directory.

Value

Bioconductor experiment containing the relative metabolite annotations stored in the feature meta-data dataframe.

Examples

```
utils::str(formals(addFMP10Annotations))
# addFMP10Annotations(spamtp, only.fmp.adduct = FALSE)
```

addSMImage	<i>Attach and optionally align an optical image to MSI</i>
------------	--

Description

Attach and optionally align an optical image to MSI

Usage

```
addSMImage(
  image_path,
  SpaMTP,
  imageId = "optical",
  sampleId = NULL,
  scaleFactor = 1,
  transformation = NULL,
  interactive = TRUE,
  ...
)
```

Arguments

image_path	Local image path.
SpaMTP	A SpatialExperiment.
imageId	Identifier for the attached image.
sampleId	Sample identifier; required for multi-sample objects.
scaleFactor	Image pixels per spatial coordinate unit.
transformation	Optional 3 by 3 matrix mapping MSI to image coordinates in full spatial units.
interactive	Open the manual alignment app when no matrix is supplied.
...	Additional arguments to shiny::runApp when interactive is TRUE.

Value

A SpatialExperiment with an imgData entry and optionally transformed spatialCoords. Coordinate alignment requires one tissue sample.

addSpatialImage	<i>Add an image using SpatialExperiment infrastructure</i>
-----------------	--

Description

This is a small SpaMTP generic around `SpatialExperiment::addImg()`. Images are stored in `imgData()` and may remain path-backed by setting `load = FALSE`.

Usage

```
addSpatialImage(
  x,
  imageSource,
  scaleFactor = 1,
  sampleId = unique(x$sample_id)[[1L]],
  imageId = tools::file_path_sans_ext(basename(imageSource)),
  load = FALSE,
  ...
)

## S4 method for signature 'SpatialExperiment'
addSpatialImage(
  x,
  imageSource,
  scaleFactor = 1,
  sampleId = unique(x$sample_id)[[1L]],
  imageId = tools::file_path_sans_ext(basename(imageSource)),
  load = FALSE,
  ...
)
```

Arguments

<code>x</code>	A <code>SpatialExperiment</code> .
<code>imageSource</code>	Local PNG/JPEG path or supported URL.
<code>scaleFactor</code>	Image scale factor.
<code>sampleId</code>	Sample identifier represented by the image.
<code>imageId</code>	Image identifier.
<code>load</code>	Whether to load the raster immediately.
<code>...</code>	Reserved for class-specific methods.

Value

The updated object.

Examples

```
methods::showMethods("addSpatialImage")
```

addSSCAnnotation	<i>Adds Cardinal ssc segmentation annotation to m/z count data object</i>
------------------	---

Description

Adds spatial shrunken centroid (SSC) labels to `Cardinal::pixelData(data)[["ssc"]]` through its public replacement accessor. This function accepts Cardinal containers only; it does not modify S4 slots directly or require Seurat.

Usage

```
addSSCAnnotation(data, data_ssc, resolution)
```

Arguments

<code>data</code>	A Cardinal Object containing the raw/binning m/z count data.
<code>data_ssc</code>	A Cardinal Object containing the ssc segmentation results. Note: Cardinal's <code>spatialShrunkenCentroids()</code> must be run to generate this object.
<code>resolution</code>	An integer (< v3.6.0) or string (>= v3.6.0) defining the ssc segmentation resolution to add to the raw Cardinal object.

Value

A Cardinal Object containing the raw m/z counts and the relative ssc segmentation annotations per pixel.

Examples

```
utils::str(formals(addSSCAnnotation))
# ssc_data <- Cardinal::spatialShrunkenCentroids(CardinalObj, ...)
# new_CardinalObj <- addSSCAnnotation(CardinalObj, ssc_data, resolution = "r=2,k=8,s=32")
```

addTranscriptome	<i>Store a paired transcriptome as an alternative experiment</i>
------------------	--

Description

The transcriptome must contain the same spatial pixels as the metabolomics object. It is stored with `SingleCellExperiment::altExp()` rather than as a second Seurat assay.

Usage

```
addTranscriptome(
  x,
  transcriptome,
  name = "transcriptome",
  assayName = "counts",
  ...
)

## S4 method for signature 'SpatialExperiment'
addTranscriptome(
  x,
  transcriptome,
  name = "transcriptome",
  assayName = "counts",
  ...
)
```

Arguments

<code>x</code>	A <code>SpatialExperiment</code> containing the primary MSI experiment.
<code>transcriptome</code>	A matrix or <code>SummarizedExperiment</code> with matching columns.
<code>name</code>	Name used in <code>altExpNames(x)</code> .
<code>assayName</code>	Assay name used when transcriptome is a matrix.
<code>...</code>	Reserved for class-specific methods.

Value

The updated `SpatialExperiment`.

Examples

```
methods::showMethods("addTranscriptome")
```

`adduct_file`*Common adduct constants*

Description

A small table of legacy adduct definitions retained for compatibility with earlier SpaMTP workflows. New annotation code should use `adductRules()`, which includes charge, stoichiometry, ion-mode, and chemical-validity fields.

Usage

```
data(adduct_file)
```

Format

A data frame with 47 rows and 6 variables:

adduct_name Adduct notation.

ion.mass Legacy ion-mass expression.

charge Ion charge.

mult Analyte stoichiometric multiplier.

add_mass Exact mass shift.

pol Ion polarity.

Value

A data frame of legacy adduct definitions.

See Also

`adductRules()`, `malDiMatrixRules()`

`adductRules`*Return the validated SpaMTP adduct rule table*

Description

The rules use neutral monoisotopic mass and explicit ion masses. Each row stores the molecular multiplier, total numerator mass shift, charge, number of protons removed from the analyte, and the charge contributed by added species. This allows rules to be checked before candidate generation.

Usage

```
adductRules(  
  polarity = c("both", "positive", "negative", "neutral"),  
  include_complex = TRUE  
)
```

Arguments

polarity One of "both", "positive", "negative", or "neutral".

include_complex Include multimers, solvent clusters, metal-exchange, and multiply charged ions.

Value

A data frame containing validated adduct rules.

Examples

```
utils::str(formals(adductRules))
```

alignSpatialOmics	<i>Align two spatial experiments using an affine transformation</i>
-------------------	---

Description

A supplied homogeneous matrix is applied reproducibly without a GUI. With transformation = NULL, an interactive preview allows translation, rotation, scaling and reflection. Coordinates are always stored in their full spatial units; an imgData scaleFactor affects only display of the reference image.

Usage

```
alignSpatialOmics(  
  sm.data,  
  st.data,  
  transformation = NULL,  
  image.slice = NULL,  
  msi.pixel.multiplier = 1,  
  shiny.host = "127.0.0.1",  
  shiny.port = NULL,  
  verbose = FALSE  
)
```

Arguments

<code>sm.data, st.data</code>	Source and reference SpatialExperiment objects, each containing one tissue sample.
<code>transformation</code>	A 3 by 3 homogeneous matrix mapping source coordinates to reference coordinates, or NULL for interactive alignment.
<code>image.slice</code>	Optional reference image ID in <code>imgData(st.data)</code> .
<code>msi.pixel.multiplier</code>	Explicit initial scaling of source coordinates. Defaults to one. Applied before transformation, including GUI transforms.
<code>shiny.host, shiny.port</code>	Host and optional port for the interactive app.
<code>verbose</code>	Show alignment messages.

Value

The source SpatialExperiment with updated spatialCoords and alignment provenance recorded by `applySpatialAlignment`.

`analyzeSpaMTPRegions` *Add region summaries and native spatial analyses to a workflow result*

Description

Adds descriptive region-versus-rest effects, detection fractions, region means and eligibility audits. Existing biological-replicate contrasts remain unchanged. Optional native spatial modules call `findSpatiallyVariableMetabolites` and `runSpatialGraphPCA` on an explicitly bounded, recorded feature/point screen. Works with any modality names, species and observation metadata, including results saved by earlier workflow versions.

Usage

```
analyzeSpaMTPRegions(
  result,
  regions = NULL,
  min_region_size = 3,
  native = list(),
  markers = TRUE,
  spatial_blocks = 6
)
```

Arguments

result	A spamtp_workflow result.
regions	A colData column for regions; NULL uses the workflow group, or workflow_cluster if clustering was requested. No labels are invented.
min_region_size	Minimum observations in both a region and its reference.
native	Named settings: moran, graph_pca, max_points (600), max_features (50), neighbors (4), lambda (0.5). These additional bounded screens default FALSE; the workflow structure module runs full-observation Graph PCA separately and uses it in the comparative analysis.
markers	Compute native scan marker scores through findAllDEMs.
spatial_blocks	Number of coordinate-only blocks per sample for marker sensitivity; zero disables block omission. These are not biological replicates.

Details

Region effects use all labelled retained observations. Blank/NA labels are excluded and counted in the audit. No P values are inferred from pixels or random pools. Moran's I statistics are exploratory and use BH within the selected feature family per modality and sample; this screening is not a genome-wide or biological-replicate test. Marker reference means equally weight other eligible regions, following scan pairwise summaries.

Value

The result with region_analysis, native_analysis and updated provenance.

Examples

```
utils::str(formals(analyzeSpaMTPRegions))
```

annotateBigData	<i>Annotates vector of m/z values</i>
-----------------	---------------------------------------

Description

This function assigns each valid m/z peak with one/multiple metabolite names based on the mass difference between the observed value and the theoretical value documented in the reference database. This function is to be used when dealing with large datasets as a preprocessing step. Users can annotate m/z values first and then subset their data accordingly before loading it into a Bioconductor experiment.

Usage

```

annotateBigData(
  mzs,
  db = NULL,
  ppm_error = NULL,
  adducts = NULL,
  polarity = NULL,
  tof_resolution = 30000,
  verbose = TRUE,
  maldi_matrix = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  ...
)

```

Arguments

<code>mzs</code>	Vector containing m/z values for annotation.
<code>db</code>	Reference metabolite dataset in the form of a data.frame. SpaMTP provides four pre-cleaned databases (HMDB_db, Lipidmaps_db, Chebi_db, GNPS_db). May be NULL when a pre-built index is supplied through ...
<code>ppm_error</code>	Mass tolerance in ppm. If NULL, a strict 5 ppm maximum is used (or a smaller value inferred from <code>tof_resolution</code>). Set to zero for exact numerical matches.
<code>adducts</code>	Optional adduct names/notations; see <code>adductRules()</code> . If NULL, use the complete rule space selected from <code>malDI_matrix</code> , or all validated general rules for the selected polarity when no matrix is given.
<code>polarity</code>	Character string defining the ion mode. When NULL, infer it from a supplied index or MALDI matrix profile, otherwise use positive.
<code>tof_resolution</code>	Instrument resolving power retained for compatibility; it can only tighten, not widen, the default 5 ppm mass-accuracy threshold.
<code>verbose</code>	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
<code>malDI_matrix</code>	Optional MALDI matrix/reagent profile used for automatic rule selection. The <code>adducts</code> argument remains an optional restriction.
<code>database_version</code>	SpaMTPdb/RaMP version used when <code>db = NULL</code> .
<code>database_source</code>	Database source used when <code>db = NULL</code> ; see loadSpaMTPDatabase() .
<code>database_local_dir</code>	Optional staged SpaMTPdb resource directory.
<code>...</code>	Additional indexed annotation/scoring arguments passed to <code>annotateTable()</code> , such as <code>index</code> , <code>rules</code> , or <code>ms1_spectrum</code> .

Value

A data.frame containing all successfully annotated m/z values, with their corresponding annotation.

Examples

```
utils::str(formals(annotateBigData))
#cardinal <- readImzML("../Test_Data/Spotted/test_data1")
#mzs <- data.frame(Cardinal::featureData(cardinal))$mz
#results <- annotateBigData(mzs, db = HMDB_db, ppm_error = 3, adducts = c("M-H", "M+Cl"), polarity = "negative")
#cardinal_subset <- Cardinal::subset(cardinal, mz %in% results$observed_mz)
```

annotateGeneIdentifiers

Store gene mapping in a native experiment's row metadata

Description

Adds aligned mapping columns to the selected experiment's `rowData()` and provenance to its `metadata()`. Expression values, feature names, row order, spatial coordinates and paired experiments are preserved. Multiple assay rows that map to one gene are annotated, not silently summed; pathway enrichment performs gene-level deduplication separately.

Usage

```
annotateGeneIdentifiers(
  object,
  assay = "main",
  id_column = NULL,
  id_type = c("auto", "symbol", "entrez", "ensembl", "uniprot", "hgnc"),
  ...
)
```

Arguments

<code>object</code>	A <code>SummarizedExperiment</code> , <code>SingleCellExperiment</code> or <code>SpatialExperiment</code> , including native objects provided by <code>SpaMTPData</code> .
<code>assay</code>	The primary experiment ("main") or transcriptome altExp name.
<code>id_column</code>	Gene identifier column in the selected <code>rowData()</code> . <code>NULL</code> uses its row names.
<code>id_type</code>	Identifier type. "auto" accepts the same input formats as mapGeneIdentifiers() . Other values explicitly prefix unqualified IDs.
<code>...</code>	Passed to mapGeneIdentifiers() , including organism, index, database and versioned reference settings.

Value

The input object with `spamtp_gene_*` mapping columns and `metadata(experiment)$gene_mapping` provenance and input audit.

Examples

```
utils::str(formals(annotateGeneIdentifiers))
```

annotateMZ	<i>Annotate one or more observed m/z values</i>
------------	---

Description

Convenience wrapper that builds an index when one is not supplied. Reuse a pre-built index for repeated or large searches.

Usage

```
annotateMZ(  
  observed_mz,  
  db = NULL,  
  index = NULL,  
  polarity = NULL,  
  adducts = NULL,  
  rules = NULL,  
  maldi_matrix = NULL,  
  ppm = 5,  
  ms1_spectrum = NULL,  
  database_version = "latest",  
  database_source = c("auto", "spamtpdb"),  
  database_local_dir = NULL,  
  infer_structure = c("auto", "never", "always"),  
  structure_backend = c("auto", "native"),  
  structure_workers = getOption("SpaMTP.smiles_workers", 1L),  
  min_structure_score = 0.05,  
  ...  
)
```

Arguments

observed_mz	Numeric vector of observed m/z values.
db	Metabolite database used when index is NULL.
index	Optional pre-built spamtp_mz_index.
polarity	Ion mode. When NULL, infer it from index or the matrix profile, falling back to positive mode.
adducts	Optional adduct subset. When NULL, retain the complete rule space selected by the matrix profile or polarity.
rules	Optional custom rule table.
malDI_matrix	Optional MALDI matrix/reagent profile used to select rules automatically when rules is NULL.
ppm	Mass tolerance in ppm.
ms1_spectrum	Optional contextual spectrum.

`database_version`
 SpaMTPdb/RaMP version used when `db = NULL`.
`database_source`
 Database source used when `db = NULL`; see [loadSpaMTPDatabase\(\)](#).
`database_local_dir`
 Optional staged SpaMTPdb resource directory.
`infer_structure,` `structure_backend,` `structure_workers,`
`min_structure_score`
 Structure-aware rule-selection arguments passed to [buildMZAnnotationIndex\(\)](#).
`...`
 Additional arguments passed to [queryMZAnnotationIndex\(\)](#).

Value

A ranked candidate data frame.

Examples

```
utils::str(formals(annotateMZ))
```

annotateSM	<i>Annotates m/z values stored in a SpaMTP Object</i>
------------	---

Description

This function assigns each valid m/z peak with one/multiple metabolite names based on the mass difference between the observed value and the theoretical value documented in the reference database. Versioned RaMP, HMDB, LIPID MAPS, ChEBI, and GNPS resources are provided by SpaMTPdb and can be combined with user-supplied reference tables.

Usage

```

annotateSM(
  data,
  db = NULL,
  assay = "main",
  raw.mz.column = "raw_mz",
  ppm_error = NULL,
  adducts = NULL,
  polarity = NULL,
  tof_resolution = 30000,
  filepath = NULL,
  return.only.annotated = TRUE,
  save.intermediate = TRUE,
  min_score = 0,
  verbose = TRUE,
  maldi_matrix = NULL,
  database_version = "latest",

```

```

    database_source = c("auto", "spamtpdb"),
    database_local_dir = NULL,
    ...
)

```

Arguments

<code>data</code>	Bioconductor experiment containing m/z values for annotation.
<code>db</code>	Reference metabolite dataset in the form of a data.frame. When NULL, the versioned SpaMTPdb chem_props table is used, unless a pre-built index is supplied through ... Versioned resources are loaded from SpaMTPdb and may be staged locally for offline use.
<code>assay</code>	Character string defining the primary or alternative experiment which contains the mz counts being annotated (default = "Spatial").
<code>raw.mz.column</code>	Character string naming the feature-metadata column containing raw m/z values without the mz- prefix. This is created by cardinalToSpatialExperiment() (default = "raw_mz").
<code>ppm_error</code>	Mass tolerance in ppm. If NULL, a strict 5 ppm maximum is used (or a smaller value inferred from tof_resolution). Set to zero for exact numerical matches.
<code>adducts</code>	Optional adduct names/notations; see adductRules(). If NULL, use the complete rule space selected from maldi_matrix, or all validated general rules for the selected polarity when no matrix is given.
<code>polarity</code>	Character string defining the ion mode. When NULL, use the MALDI matrix profile default, a supplied index's mode, or positive mode when neither is available.
<code>tof_resolution</code>	Instrument resolving power retained for compatibility; it can only tighten, not widen, the default 5 ppm mass-accuracy threshold.
<code>filepath</code>	Character string of the directory to store the _annotated_mz_peaks.csv. If set to NULL no dataframe will be saved (default = NULL).
<code>return.only.annotated</code>	Boolean value indicating if the annotated Bioconductor experiment should only include m/z values that were successfully annotated (default = TRUE).
<code>save.intermediate</code>	Boolean indicating whether to store the scored annotation result and its pipeline/RaMP provenance in S4Vectors::metadata(data)\$mz_annotation. A compatibility copy is retained under the db_3 key (default = TRUE).
<code>min_score</code>	Minimum annotation score retained in the stored candidate table. The default 0 keeps all ppm-valid candidates so downstream pathway functions can apply a user-defined threshold without re-running annotation.
<code>verbose</code>	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
<code>malDI_matrix</code>	Optional MALDI matrix or derivatization reagent name. When supplied, SpaMTP selects validated matrix-specific rules automatically. adducts is optional and only restricts that automatic search space when explicitly supplied.
<code>database_version</code>	SpaMTPdb/RaMP version used when db = NULL.

database_source	Database source used when db = NULL; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
...	Additional indexed annotation/scoring arguments passed to annotateTable() , such as index, rules, ms1_spectrum, infer_structure, structure_backend, structure_workers, or min_structure_score.

Value

A Bioconductor experiment with m/z values annotated. These annotations are stored in the selected experiment's `rowData()`.

Examples

```
utils::str(formals(annotateSM))
# HMDB_db <- load("data/HMDB_1_names.rds")
```

annotateSMILESStructure

Add SMILES-derived structural features to a metabolite database

Description

Add SMILES-derived structural features to a metabolite database

Usage

```
annotateSMILESStructure(
  db,
  smiles_column = NULL,
  backend = c("auto", "native"),
  overwrite = FALSE,
  strict = FALSE,
  workers = getOption("SpaMTP.smiles_workers", 1L)
)
```

Arguments

db	A metabolite data frame.
smiles_column	Column containing SMILES. When NULL, SpaMTP detects iso_smiles, canonical_smiles, or smiles.
backend	Parser backend passed to deconvolveSMILES() .
overwrite	Replace existing feature values. By default only absent or missing values are filled.
strict	Passed to deconvolveSMILES() .
workers	Parallel workers passed to deconvolveSMILES() .

Value

db with structure-derived columns appended or completed.

Examples

```
utils::str(formals(annotateSMILESstructure))
```

annotateTable	<i>Annotates m/z values stored in a data.frame based on reference metabolite dataset</i>
---------------	--

Description

Helper function for annotateSM() and fishersPathwayAnalysis().

Usage

```
annotateTable(  
  mz_df,  
  db = NULL,  
  ppm_error = NULL,  
  adducts = NULL,  
  polarity = NULL,  
  tof_resolution = 30000,  
  verbose = TRUE,  
  index = NULL,  
  rules = NULL,  
  ms1_spectrum = NULL,  
  use_mass_defect = TRUE,  
  check_isotopes = TRUE,  
  check_adduct_network = TRUE,  
  min_score = 0,  
  maldi_matrix = NULL,  
  database_version = "latest",  
  database_source = c("auto", "spamtpdb"),  
  database_local_dir = NULL,  
  infer_structure = c("auto", "never", "always"),  
  structure_backend = c("auto", "native"),  
  structure_workers = getOption("SpaMTP.smiles_workers", 1L),  
  min_structure_score = 0.05  
)
```

Arguments

mz_df	dataframe containing m/z values for annotation.
db	Reference metabolite dataset in the form of a data.frame. May be NULL when index is supplied.

ppm_error	Mass tolerance in ppm. If NULL, a strict 5 ppm maximum is used (or a smaller value inferred from tof_resolution). Set to zero for exact numerical matches.
adducts	Optional adduct names/notations; see adductRules(). If NULL, use the complete rule space selected from maldi_matrix, or all validated general rules for the selected polarity when no matrix is given.
polarity	Character string defining the ion mode. When NULL, infer it from a supplied index or MALDI matrix profile, otherwise use positive.
tof_resolution	Instrument resolving power retained for compatibility; it can only tighten, not widen, the default 5 ppm mass-accuracy threshold.
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
index	Optional reusable index created by buildMZAnnotationIndex().
rules	Optional custom adduct rule table; see adductRules().
ms1_spectrum	Optional contextual MS1 spectrum with mz and intensity columns for isotope and adduct-family scoring.
use_mass_defect	Apply the CHO negative mass-defect penalty.
check_isotopes	Score carbon-13 and halogen isotope evidence when a contextual spectrum is supplied.
check_adduct_network	Downweight complex ions whose base monomer family is absent from the contextual spectrum.
min_score	Minimum final annotation score to retain.
malDI_matrix	Optional MALDI matrix/reagent profile used for automatic rule selection. adducts = NULL keeps the complete selected rule space.
database_version	SpaMTPdb/RaMP version used when db = NULL.
database_source	Database source used when db = NULL; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
infer_structure	"auto" joins precomputed SpaMTPdb structure features when possible, "never" disables inference, and "always" parses missing structures at runtime.
structure_backend	SMILES parser passed to deconvolveSMILES().
structure_workers	Number of workers used for runtime SMILES parsing.
min_structure_score	Minimum rule-specific structural prior retained during candidate-index construction.

Value

Generates an intermediate annotated m/z dataframe

Examples

```
### HelperFunction
```

annotationInfo	<i>Inspect the metabolite annotation provenance stored in a SpaMTP object</i>
----------------	---

Description

Reports whether downstream pathway functions will use the current indexed, scored RaMP annotation pipeline or a legacy mass-match result.

Usage

```
annotationInfo(SpaMTP)
```

Arguments

SpaMTP	A SingleCellExperiment, including SpatialExperiment.
--------	--

Value

A named list containing annotation schema, engine, RaMP version, provenance, and candidate count where available.

Examples

```
utils::str(formals(annotationInfo))
```

applySpatialAlignment	<i>Apply automatic or precomputed spatial alignment</i>
-----------------------	---

Description

Aligns a moving Spatial Metabolomics (SM) SpatialExperiment to a fixed SpatialExperiment. The function can apply coordinates exported by the SMINT workflow, apply a homogeneous transformation matrix, estimate an affine transform from paired landmarks, or run the STalign LDDMM backend used by SMINT through reticulate.

Usage

```

applySpatialAlignment(
  SM.data,
  ST.data = NULL,
  alignment = NULL,
  method = c("lddmm", "affine"),
  SM.fov = NULL,
  ST.image = NULL,
  SM.boundary = "centroids",
  SM.landmarks = NULL,
  ST.landmarks = NULL,
  landmark.order = c("xy", "yx"),
  SM.landmark.space = c("original", "preprocessed"),
  coordinate.columns = NULL,
  cell.column = NULL,
  ST.scale.factor = NULL,
  source.scale = 1,
  source.rotate = 0,
  source.flip = c(FALSE, FALSE),
  source.translate = c(0, 0),
  source.origin = NULL,
  dx = 30,
  niter = 1000,
  epV = 200,
  device = c("auto", "cpu", "cuda"),
  seed = 1,
  lddmm.args = list(),
  python = NULL,
  alignment.name = "SMINT",
  store = TRUE,
  return = c("object", "result"),
  diagnostics.max.points = 10000,
  verbose = TRUE
)

```

Arguments

SM.data	A SpatialExperiment containing the moving SM coordinates.
ST.data	An optional matching spatial container with fixed coordinates. It is required when computing an affine or LDDMM alignment, but is optional when applying final coordinates supplied through alignment.
alignment	Optional precomputed alignment. Accepted inputs are a data frame or matrix of final coordinates; a 2-by-3 or 3-by-3 homogeneous transformation matrix; a list containing aligned_coordinates, coordinates, transformation, or matrix; a CSV/TSV/RDS/JSON file; or a SMINT output directory containing aligned_coordinates.csv and optionally transformation.json. Coordinate columns produced by the existing SpaMTP and SMINT notebooks

	(x_transformed, x_new, and x_final, with their y counterparts) are detected automatically.
method	Alignment method used when alignment is NULL. "affine" estimates a six-degree-of-freedom transformation from paired landmarks in R. "lddmm" runs STalign's landmark-guided affine plus diffeomorphic registration through Python.
SM.fov	Optional imgData image ID for the moving experiment. By default, the first image in SM.data is used.
ST.image	Optional imgData image ID for the fixed experiment. By default, the first image in ST.data is used.
SM.boundary	Compatibility label; coordinates are always read from spatialCoords().
SM.landmarks, ST.landmarks	Matched landmark coordinates as two-column matrices or data frames in x-y order. At least three non-collinear pairs are required for method = "affine". Landmarks are optional but strongly recommended for method = "lddmm" when the sections are not already close. ST landmarks must use the target coordinate system after any ST.scale.factor has been applied.
landmark.order	Coordinate-column order in both landmark inputs. "xy" is the R/SpaMTP convention. Use "yx" for row-column arrays saved by the notebook's point annotator.
SM.landmark.space	Whether SM landmarks use the "original" FOV coordinates or the "preprocessed" coordinates after source scaling, rotation, reflection, translation, and origin adjustment. Point-annotator landmarks made from the rasterized notebook output are preprocessed.
coordinate.columns	Optional names or integer positions of the x and y columns in a precomputed coordinate table.
cell.column	Optional name or integer position of the cell/spot ID column in a precomputed coordinate table. When no IDs are available, rows must already be in the same order as the moving SM centroids.
ST.scale.factor	Optional fixed-coordinate scale factor. Supply a numeric value or a scale-factor name such as "hires" or "lowres" for Visium images. NULL uses full-resolution coordinates.
source.scale	Numeric scalar or x-y vector applied to moving coordinates before fitting. This reproduces the initial SM scaling step in the SMINT notebooks.
source.rotate	Counter-clockwise rotation in degrees, around the origin, applied after scaling.
source.flip	Logical x-y vector indicating reflection around the origin.
source.translate	Numeric x-y translation applied after scaling, reflection, and rotation.
source.origin	Optional numeric x-y lower bound. Coordinates are shifted only where necessary so that their minima are at least this value. Use c(0, 0) to reproduce the non-negative coordinate step in the notebook.
dx	Raster spacing passed to STalign.rasterize().

niter	Number of LDDMM optimisation iterations.
epV	Velocity-field gradient step passed to STalign.LDDMM().
device	One of "auto", "cpu", or "cuda". "auto" uses CUDA when Torch reports that it is available.
seed	Random seed set for NumPy and Torch before LDDMM.
lddmm.args	Named list of additional arguments passed to STalign.LDDMM(). Values here override niter and epV.
python	Optional path to a Python executable containing STalign and torch. It must be selected before reticulate initialises Python.
alignment.name	Name used to store alignment provenance in S4Vectors::metadata(SM.data)\$spatial_alignment.
store	Logical; store lightweight provenance and diagnostics in the returned object.
return	Either "object" (default) or "result". The latter returns a list containing the object, coordinate tables, diagnostics, parameters, and lightweight backend information.
diagnostics.max.points	Maximum number of moving and fixed points used for nearest-neighbour diagnostics.
verbose	Logical; print progress messages.

Details

Python is only required when `method = "lddmm"` and `alignment = NULL`. Precomputed SMINT coordinates and affine landmark alignment are handled entirely in R. The backend summary records STalign's affine component as `affine_yx`; it is not the complete nonlinear LDDMM transformation.

Value

The updated spatial container when `return = "object"`; otherwise an object of class `spamtp_spatial_alignment` containing the aligned object and alignment details.

Examples

```
utils::str(formals(applySpatialAlignment))
# Apply coordinates already exported by the SMINT/STalign notebook.
aligned_sm <- applySpatialAlignment(
  SM.data = sm,
  ST.data = st,
  alignment = "Ven5_z2_transformed_metabolites_coordinates.csv",
  coordinate.columns = c("x_transformed", "y_transformed")
)

# Run the notebook's landmark-guided LDDMM workflow from R.
fit <- applySpatialAlignment(
  SM.data = sm,
  ST.data = st,
  method = "lddmm",
  SM.landmarks = sm_landmarks,
```

```

    ST.landmarks = st_landmarks,
    landmark.order = "yx",          # point-annotator row/column order
    SM.landmark.space = "preprocessed", # landmarks selected after rasterization
    source.scale = 10,
    source.rotate = 90,
    source.origin = c(0, 0),
    python = "/path/to/STalign_env/bin/python",
    return = "result"
)
fit$diagnostics

```

asCardinal

*Convert a SpatialExperiment to a Cardinal experiment***Description**

Convert a SpatialExperiment to a Cardinal experiment

Usage

```

asCardinal(x, ...)

## S4 method for signature 'SpatialExperiment'
asCardinal(x, ...)

## S4 method for signature 'MSImagingExperiment'
asCardinal(x, ...)

## S4 method for signature 'MSImagingArrays'
asCardinal(x, ...)

```

Arguments

x An object to convert.

... Arguments passed to the class-specific method.

Value

A Cardinal MSImagingExperiment, or unchanged MSImagingArrays input.

Examples

```
methods::showMethods("asCardinal")
```

asSpatialExperiment	<i>Convert an object to a SpatialExperiment</i>
---------------------	---

Description

asSpatialExperiment() is the Bioconductor-native conversion entry point used by SpaMTP. Continuous or unaligned raw spectra should remain in a Cardinal MSImagingArrays or MSImagingExperiment; conversion is intended for data with a common feature-by-pixel matrix, typically after peak alignment or binning.

Usage

```
asSpatialExperiment(x, ...)

## S4 method for signature 'MSImagingExperiment'
asSpatialExperiment(x, ...)

## S4 method for signature 'MSImagingArrays'
asSpatialExperiment(x, ...)

## S4 method for signature 'SpatialExperiment'
asSpatialExperiment(x, ...)

## S4 method for signature 'SummarizedExperiment'
asSpatialExperiment(x, coordinates = c("x", "y"), ...)
```

Arguments

x	An object to convert.
...	Arguments passed to the class-specific method.
coordinates	Column names in colData(x) containing x/y coordinates when converting a SummarizedExperiment.

Value

A SpatialExperiment.

Examples

```
methods::showMethods("asSpatialExperiment")
```

binMetabolites	<i>Sum selected metabolite features per pixel</i>
----------------	---

Description

The sum is stored in colData without modifying the expression assays.

Usage

```
binMetabolites(  
  data,  
  mzs,  
  assay = "main",  
  slot = "counts",  
  bin_name = "Binned_Metabolites"  
)
```

Arguments

data	A SpatialExperiment or aligned Cardinal experiment.
mzs	Character feature IDs to sum. Duplicates count once.
assay	Primary or alternative experiment.
slot	Expression assay. See experimentAccess for the experiment/matrix distinction.
bin_name	Output colData column name. Existing columns are protected.

Value

The updated SpatialExperiment.

Examples

```
utils::str(formals(binMetabolites))
```

binnedCardinalToSeurat	<i>Convert a binned Cardinal matrix to an optional Seurat object</i>
------------------------	--

Description

Convert a binned Cardinal matrix to an optional Seurat object

Usage

```

binnedCardinalToSeurat(
  data,
  mtx,
  multi.run = FALSE,
  assay = "Spatial",
  verbose = TRUE
)

```

Arguments

<code>data</code>	Cardinal object supplying coordinates and metadata.
<code>mtx</code>	Binned feature-by-pixel matrix.
<code>multi.run</code>	Retained for compatibility.
<code>assay</code>	Name of the Seurat assay.
<code>verbose</code>	Show conversion messages.

Value

A Seurat object.

Examples

```
utils::str(formals(binnedCardinalToSeurat))
```

binSpaMTP

Bin aligned spatial metabolomics features

Description

MSImagingArrays or MSImagingExperiment input delegates binning to Cardinal and is converted to SpatialExperiment only after a common feature axis exists. SpatialExperiment input returns a new binned SpatialExperiment. For aligned matrices, bins start at the minimum observed m/z and use linear (m/z) or log-mass (ppm) spacing. Each feature is assigned once to its nearest bin on that axis (midpoints go up); empty bins are omitted. Thus sum binning conserves each pixel's total intensity. Cardinal input uses Cardinal's reference-window binning convention instead.

Usage

```

binSpaMTP(
  data,
  resolution,
  units = "ppm",
  assay = "main",
  slot = "counts",
  method = c("sum", "mean", "max", "min"),

```

```
    return.only.mtx = FALSE
  )

## S4 method for signature 'SpatialExperiment'
binSpaMTP(
  data,
  resolution,
  units = "ppm",
  assay = "main",
  slot = "counts",
  method = c("sum", "mean", "max", "min"),
  return.only.mtx = FALSE
)

## S4 method for signature 'MSImagingExperiment'
binSpaMTP(
  data,
  resolution,
  units = "ppm",
  assay = "main",
  slot = "counts",
  method = c("sum", "mean", "max", "min"),
  return.only.mtx = FALSE
)

## S4 method for signature 'MSImagingArrays'
binSpaMTP(
  data,
  resolution,
  units = "ppm",
  assay = "main",
  slot = "counts",
  method = c("sum", "mean", "max", "min"),
  return.only.mtx = FALSE
)

## S4 method for signature 'ANY'
binSpaMTP(
  data,
  resolution,
  units = "ppm",
  assay = "main",
  slot = "counts",
  method = c("sum", "mean", "max", "min"),
  return.only.mtx = FALSE
)
```

Arguments

data	A SpatialExperiment, Cardinal MSImagingArrays or MSImagingExperiment.
resolution	Positive bin resolution.
units	Resolution units, either "ppm" or "mz".
assay	Primary MSI experiment name; alternative modalities cannot be binned.
slot	Matrix layer or SpatialExperiment assay to bin. See experimentAccess for the experiment/matrix distinction.
method	Aggregation method.
return.only.mtx	Return only the binned matrix.

Value

A binned SpatialExperiment or matrix.

Examples

```
methods::showMethods("binSpaMTP")
```

buildGeneMappingIndex *Build a gene identity index from RaMP and HGNC*

Description

Connects RaMP gene records to HGNC using explicit stable identifiers, then approved and previous symbols. Shared aliases are never used to merge RaMP records. Conflicting stable identifiers quarantine the affected RaMP record. Multiple non-conflicting RaMP records for one HGNC gene share one canonical RaMP ID (lexicographically first); their pathway memberships can then be united and counted once. No transitive merging across different HGNC genes or orthology conversion is performed. A fingerprint of normalized RaMP identifiers and common names guards index reuse. Reordering rows, duplicating evidence or changing unrelated columns does not invalidate it. Rebuild saved indices from older schema versions with this function before using them in a new workflow.

Usage

```
buildGeneMappingIndex(
  source_df,
  gene_reference = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  offline = FALSE
)
```

Arguments

source_df	RaMP identifier table from SpaMTPdb, with sourceId, rampId and optionally commonName.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spaMTPdbGeneRef . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
offline	Require local or cached HGNC data.

Value

A spamtp_gene_index list containing reference lookup, gene groups, per-RaMP identity decisions, precomputed conflict membership, evidence and resource provenance including source_fingerprint.

Examples

```
utils::str(formals(buildGeneMappingIndex))
```

```
buildMZAnnotationIndex
```

Build a reusable indexed metabolite annotation search space

Description

Expected adduct m/z values are generated once, sorted, and queried by binary interval search. This is the one-dimensional equivalent of an interval tree and avoids scanning every database row for every observed peak.

Usage

```
buildMZAnnotationIndex(
  db,
  polarity = NULL,
  adducts = NULL,
  rules = NULL,
  maldi_matrix = NULL,
  collapse_isomers = TRUE,
  infer_structure = c("auto", "never", "always"),
  structure_backend = c("auto", "native"),
  structure_workers = getOption("SpaMTP.smiles_workers", 1L),
  min_structure_score = 0.05
)
```

Arguments

db	Metabolite database. Both the legacy SpaMTP five-column/wide database and the RaMP chem_props schema are supported.
polarity	"positive", "negative", or "neutral". When NULL, use the matrix-profile default, or positive mode when no profile is given.
adducts	Optional character vector of adduct names or bracketed notations. When NULL, use the complete automatically selected rule space; this is a filter, not a compulsory input.
rules	Optional custom rule table. See adductRules() .
maldi_matrix	Optional MALDI matrix or derivatization reagent profile. When supplied and rules is NULL, maldiMatrixRules() automatically selects standard and validated matrix-specific rules. adducts remains an optional filter on that selected rule space.
collapse_isomers	Collapse records sharing formula, exact mass, and proton bound before indexing.
infer_structure	"auto" joins precomputed SpaMTPdb features or derives them for at most <code>getOption("SpaMTP.max_runtime")</code> unique SMILES, "never" disables inference, and "always" forces runtime parsing and replaces precomputed structural fields.
structure_backend	SMILES parser used by deconvolveSMILES() .
structure_workers	Number of workers used for runtime SMILES parsing.
min_structure_score	Minimum rule-specific structural prior retained before m/z indexing. Set to zero to rank without structure-based pruning.

Value

An object of class `spamp_mz_index`.

Examples

```
utils::str(formals(buildMZAnnotationIndex))
```

buildPathwayIndex	<i>Build a shared, auditable pathway membership index</i>
-------------------	---

Description

Unites pathway memberships of non-conflicting RaMP records belonging to one HGNC gene. Retains raw memberships and excluded conflicting records. Reuse this index in enrichment, expression aggregation, scoring and named pathway plots to obtain the same identities and membership sets. Pathways are keyed by `pathwayRampId`; identical display names never merge distinct pathways.

Usage

```
buildPathwayIndex(
  database = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL
)
```

Arguments

database	A resource bundle with analytehaspathway and pathway, and source_df for gene harmonization. NULL retrieves these from SpaMTPdb.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spamtpdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.

Value

A `spamtp_pathway_index` containing canonical and raw member sets, conflicting members, meta-data, `gene_index` and resource fingerprints. Database size means unique analyte identities after reconciliation and conflict exclusion, before intersecting with measured features.

Examples

```
db <- list(analytehaspathway = data.frame(rampId = "RAMP_C_1",
      pathwayRampId = "P1"), pathway = data.frame(pathwayRampId = "P1"))
buildPathwayIndex(db)$members
```

calculate_bounds	<i>Calculates the mz range of the observed_df</i>
------------------	---

Description

Calculates the mz range of the `observed_df`

Usage

```
calculate_bounds(input_df)
```

Arguments

<code>input_df</code>	DataFrame of the observed dataframe being annotated
-----------------------	---

Value

A list containing the lower and upper mz range for the provided sample

Examples

```
# mz_df$row_id <- seq(1, length(mz_df$mz))

# mass_range <- calculate_bounds(mz_df)
# lower_bound <- mass_range$lower_bound
# upper_bound <- mass_range$upper_bound
```

```
calculateAnnotationStatistics
```

Rank annotations for all MSI features

Description

When pathway.assay contains RaMP analytes rather than pathway scores, createPathwayObject is reused to calculate scores in a temporary altExp. Set pathway.scores=TRUE to use an existing pathway-level assay directly. Input containers are never modified.

Usage

```
calculateAnnotationStatistics(
  data,
  mz.assay = "main",
  pathway.assay = "pathway",
  mz.slot = "counts",
  pathway.slot = "counts",
  return.top = TRUE,
  corr_threshold = 0,
  corr_weight = 1,
  n_weight = 1,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  pathway.scores = FALSE,
  pathway_index = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  duplicate_genes = c("error", "mean", "sum")
)
```

Arguments

data	A SingleCellExperiment or SpatialExperiment with paired altExp.
mz.assay	Primary experiment ("main") or alternative MSI experiment.
pathway.assay	Alternative experiment of RaMP analytes, or pathway scores when pathway.scores=TRUE.
mz.slot	Expression assay in the MSI experiment. See experimentAccess for the experiment/matrix distinction.
pathway.slot	Input expression assay in pathway.assay. See experimentAccess for the experiment/matrix distinction.

return.top	Return one best evaluable candidate per feature; FALSE returns a named list of all candidate tables. Features without a score remain as NA rows in top results and NULL entries when no table is possible.
corr_theshold	Minimum signed pathway correlation (legacy spelling).
corr_weight, n_weight	Non-negative weights, not both zero.
database	Optional named list from loadSpaMTPDatabase.
database_version, database_source, database_local_dir	Database selection; see loadSpaMTPDatabase.
pathway.scores	Whether pathway.assay already contains pathway scores.
pathway_index	A reusable index from buildPathwayIndex(). When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spaMTPdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.

Details

Uses the score definition and missing-data rules documented in [calculateSingleAnnotationStatistics\(\)](#). The reported pval and pval_adj columns are heuristic ranking measures, not calibrated identification p-values or biological-replicate inference.

Value

A data.frame preserving feature order and rowData, with ranked annotation columns; or a named list when return.top=FALSE.

Examples

```
utils::str(formals(calculateAnnotationStatistics))
```

```
calculateSingleAnnotationStatistics
```

Rank candidate metabolite annotations using paired pathway expression

Description

Pearson correlations are calculated directly between a selected m/z assay and pathway expression in altExp, aligned by pixel IDs. This replaces the former temporary Seurat assay and synthetic-mass Cardinal conversion.

Usage

```
calculateSingleAnnotationStatistics(
  mz,
  data,
  mz.assay = "main",
  pathway.assay = "pathway",
  mz.slot = "counts",
  pathway.slot = "pathwayScores",
  corr_threshold = 0,
  corr_weight = 1,
  n_weight = 1,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL
)
```

Arguments

mz	One numeric m/z query or exact character feature ID.
data	A SingleCellExperiment or SpatialExperiment with paired altExp.
mz.assay	Primary experiment ("main") or alternative MSI experiment.
pathway.assay	Alternative experiment containing pathway-level scores.
mz.slot	Expression assay in the MSI experiment. See experimentAccess for the experiment/matrix distinction.

`pathway.slot` Expression assay in the pathway experiment. See [experimentAccess](#) for the experiment/matrix distinction.

`corr_theshold` Minimum signed pathway correlation (legacy spelling).

`corr_weight, n_weight`
Non-negative weights, not both zero.

`database` Optional named list from `loadSpaMTPDatabase`.

`database_version, database_source, database_local_dir`
Database selection; see `loadSpaMTPDatabase`.

Details

For each candidate, `max_cor` is the signed correlation of the pathway with largest absolute correlation; ties use pathway ID order. `n_sig_path` counts finite correlations strictly greater than `corr_theshold` (zero therefore counts positive correlations, not every pathway). The score is `corr_weight * scale(abs(max_cor)) + n_weight * scale(n_sig_path)`, across candidates for one feature. A constant component contributes zero. Constant expression and unmeasured pathways have undefined correlations. Scores require two evaluable candidates; otherwise scores and probabilities are NA.

`pval` is the standard-normal upper tail of this weighted score; `pval_adj` applies BH within each feature. These are legacy heuristic ranking measures, not calibrated identification p-values: the weighted components are not independent, the combined score need not be standard normal, and pixels are not biological replicates. Correlation is across all supplied pixels; subset samples/conditions beforehand when appropriate.

Current scored RaMP annotations take precedence over rowData RaMP IDs. Older `all_Isomers_IDs` columns can be mapped via `source_df$sourceId`. Duplicate candidate IDs are counted once.

Value

A tibble ordered by decreasing score, containing `metabolite`, `ramp_id`, `n_sig_path`, `max_cor`, `z_score`, `pval` and `pval_adj`.

Examples

```
utils::str(formals(calculateSingleAnnotationStatistics))
```

<code>cardinalToSeurat</code>	<i>Convert a Cardinal object to an optional Seurat object</i>
-------------------------------	---

Description

Convert a Cardinal object to an optional Seurat object

Usage

```
cardinalToSeurat(
  data,
  multi.run = FALSE,
  seurat.coord = NULL,
  assay = "Spatial",
  verbose = TRUE
)
```

Arguments

<code>data</code>	A Cardinal MSImagingExperiment with aligned spectra.
<code>multi.run</code>	Retained for compatibility; run identities are preserved during the intermediate conversion.
<code>seurat.coord</code>	Optional replacement coordinates with X_new and Y_new columns.
<code>assay</code>	Name of the Seurat assay.
<code>verbose</code>	Show conversion messages.

Value

A Seurat object.

Examples

```
utils::str(formals(cardinalToSeurat))
```

cardinalToSpatialExperiment

Convert an aligned Cardinal experiment to SpatialExperiment

Description

The intensity matrix is stored as an assay, Cardinal `featureData()` becomes `rowData()`, `pixelData()` becomes `colData()`, and `coord()` becomes `spatialCoords()`. This conversion materialises a matrix-backed intensity matrix; raw continuous data should be binned with Cardinal before conversion.

Usage

```
cardinalToSpatialExperiment(x, assayName = "counts", materialize = TRUE)
```

Arguments

<code>x</code>	A Cardinal MSImagingExperiment with a shared feature axis.
<code>assayName</code>	Name of the assay in the returned object.
<code>materialize</code>	Whether a non-matrix Cardinal assay may be materialised.

Value

A SpatialExperiment.

Examples

```
utils::str(formals(cardinalToSpatialExperiment))
```

check_and_truncate_adduct_vector

Checks if the complete adduct is in the data base, else returns a truncated adduct

Description

Checks if the complete adduct is in the data base, else returns a truncated adduct

Usage

```
check_and_truncate_adduct_vector(adduct, db, verbose = TRUE)
```

Arguments

adduct	Character string defining the adduct to be checked.
db	DataFrame of the current reference database.
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Value

Character string of the complete or truncated adduct

Examples

```
# HMDB_db <- load("data/HMDB_1_names.rds")  
# check_and_truncate_adduct_vector(c("M+H"), HMDB_db)
```

 check_cardinal_version

Check Cardinal Version

Description

Checks if the Cardinal Package installed is $\geq$ version 3.6.0

Usage

```
check_cardinal_version()
```

Value

Boolean indicating if the version of Cardinal is $\geq$ 3.6.0

checkAlignment

Inspect the alignment of two spatial experiments

Description

Inspect the alignment of two spatial experiments

Usage

```
checkAlignment(
  SM.data,
  ST.data,
  names = c("SM", "ST"),
  cols = NULL,
  image.slice = NULL,
  size = 0.5
)
```

Arguments

SM.data, ST.data

SpatialExperiment objects in the same coordinate units.

names

Two labels for the source and target measurements.

cols

Two point colours.

image.slice

Optional image ID in ST.data. With multiple samples, subset to one sample before drawing an image overlay.

size

Point size.

Value

A ggplot; multiple samples are shown in separate panels.

`combineTransformations`*Combines rigid tranformation matrices*

Description

Combines rigid tranformation matrices in the following order: translation of points to origin (0, 0)
-> reflection of points -> rotation by alpha degrees and translation of points to new center

Usage

```
combineTransformations(  
  center.cur,  
  center.new,  
  alpha,  
  mirror.x = FALSE,  
  mirror.y = FALSE  
)
```

Arguments

<code>center.cur</code>	(x, y) image pixel coordinates specifying the current center of the tissue (stored as the centers auxiliary entry)
<code>center.new</code>	(x, y) image pixel coordinates specifying the new center (image center)
<code>alpha</code>	Rotation angle
<code>mirror.x, mirror.y</code>	Logical values indicating reflection across the x or y axis before rotation.

Value

A 3-by-3 homogeneous affine-transformation matrix.

Examples

```
utils::str(formals(combineTransformations))  
transformation <- combineTransformations(  
  center.cur = c(0, 0),  
  center.new = c(10, 20),  
  alpha = 90  
)  
stopifnot(identical(dim(transformation), c(3L, 3L)))
```

`convertSeuratToCardinal`*Convert an optional Seurat object to Cardinal*

Description

Convert an optional Seurat object to Cardinal

Usage

```
convertSeuratToCardinal(  
  data,  
  assay = "Spatial",  
  slot = "counts",  
  run_col = NULL,  
  feature.metadata = FALSE,  
  verbose = TRUE  
)
```

Arguments

<code>data</code>	A Seurat object.
<code>assay</code>	Seurat assay to convert.
<code>slot</code>	Exact Seurat expression layer name, read with <code>SeuratObject::LayerData()</code> , not an internal S4 slot. This boundary converter differs from native analysis selectors; see experimentAccess and seuratToSpatialExperiment() .
<code>run_col</code>	Optional cell-metadata column containing run identifiers.
<code>feature.metadata</code>	Retained for compatibility. Feature metadata are always preserved.
<code>verbose</code>	Show conversion messages.

Value

A Cardinal MSImagingExperiment.

Examples

```
utils::str(formals(convertSeuratToCardinal))
```

createMergedModalityAssay

Combine paired modalities in a scaled alternative experiment

Description

Combines selected modalities as a scaled alternative experiment. Useful for integrating multiple modalities (e.g. transcriptomics, proteomics, metabolomics) that have already been scaled.

Usage

```
createMergedModalityAssay(
  SpaMTP,
  assays.to.merge,
  new.assay = "merged",
  return.original = TRUE,
  verbose = FALSE
)
```

Arguments

SpaMTP	A SpatialExperiment containing at least two paired modalities.
assays.to.merge	At least two distinct modality names: "main" and/or names in <code>SingleCellExperiment::altExpNames(SpaMTP)</code> . Multiple aliases for the same primary experiment are not distinct modalities. See experimentAccess .
new.assay	Name of the output alternative experiment, not an expression matrix (default = "merged").
return.original	TRUE adds an altExp; FALSE returns only the merged feature space.
verbose	Reserved compatibility argument; no progress messages are emitted.

Details

Prefers the native scaled assay, then the legacy `scale.data` assay; otherwise centres and scales logcounts, normcounts or counts across pixels. Constant features become zero. The merged values are stored as scaled, never relabelled as counts.

Value

If `return.original = TRUE`, the input SpatialExperiment with `altExp(SpaMTP, new.assay)` containing a scaled assay. Otherwise a new SpatialExperiment with merged rows, the same pixels and a scaled assay.

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = rbind(a = 1:4, b = 4:1)),
  spatialCoords = cbind(x = 1:4, y = 0),
  colData = S4Vectors::DataFrame(row.names = paste0("p", 1:4)))
rna <- matrix(c(1, 3, 2, 4), nrow = 1,
  dimnames = list("gene1", colnames(x)))
x <- addTranscriptome(x, rna)
x <- createMergedModalityAssay(x, c("main", "transcriptome"))
merged <- SingleCellExperiment::altExp(x, "merged")
SummarizedExperiment::assay(merged, "scaled")
```

createPathwayAssay	<i>Create a RaMP identity alternative experiment</i>
--------------------	--

Description

Maps genes or mass features to RaMP identities in a new altExp(). Rows are gene/compound identities, not pathway scores. Use this identity expression for enrichment or co-regulation analysis, or call [createPathwayObject\(\)](#) to construct pathway-level scores.

Usage

```
createPathwayAssay(
  SpaMTP,
  analyte_type = "metabolites",
  assay = "Spatial",
  slot = "counts",
  new_assay = "pathway",
  annotation_score_threshold = 0.05,
  annotation_source = c("current", "auto", "legacy"),
  verbose = TRUE,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  pathway_index = NULL,
  duplicate_genes = c("error", "mean", "sum"),
  id_column = NULL,
  metabolite_ambiguity = c("expand", "exclude")
)
```

Arguments

SpaMTP	A SingleCellExperiment, including SpatialExperiment, containing metabolomic or transcriptomic expression.
analyte_type	Character string specifying the type of analytes to process. Must be either "genes" or "metabolites" (default = "metabolites").
assay	Primary ("main") or alternative experiment name. The default "Spatial" is a compatibility alias for the primary experiment.
slot	Expression assay name within the selected experiment (default = "counts"), read with SummarizedExperiment::assay(). See experimentAccess for the experiment/matrix distinction.
new_assay	Name for the identity-level altExp() (default = "pathway" for compatibility); this is not an expression assay name.
annotation_score_threshold	Minimum indexed annotation score used to map m/z features to RaMP compounds (default = 0.05).
annotation_source	Metabolite annotation provenance. "current" requires the indexed, scored RaMP output; "auto" and "legacy" enable compatibility with older serialized SpaMTP objects.
verbose	Boolean logical value indicating whether to print verbose messages during execution. (default = TRUE).
database	Optional named list of database resources, normally created by loadSpaMTPDatabase() .
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spaMTPdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.

gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
pathway_index	A reusable index from buildPathwayIndex(). When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.
id_column	Gene identifier column in the selected rowData(). NULL uses its row names.
metabolite_ambiguity	Expand a mass feature into candidate compounds (historical behavior), or exclude features mapping to more than one compound. The workflow explicitly defaults to exclude. Candidate pairs, excluded features and exact aggregation weights are retained in metadata.

Details

Gene expression uses the shared HGNC identity index. Non-conflicting RaMP records of one gene count once. Different expression rows for a gene require an explicit aggregation choice. In custom RaMP-only compatibility mode the historical default is mean. The selected expression assay name is preserved for genes, including logcounts. Inputs, member coverage, duplicate handling and index provenance are stored in `metadata(altExp(x, new_assay))$pathway_mapping`.

Value

The input container with a new `SingleCellExperiment::altExp(SpaMTP, new_assay)`. Its expression assay retains the name supplied in `slot`, and its `rowData()` describes the RaMP identities and original measured features.

Examples

```
utils::str(formals(createPathwayAssay))
## Create a pathway assay from metabolite data
#spamtp_obj <- createPathwayAssay(spamtp_obj, analyte_type = "metabolites", assay = "SPM", new_assay = "pathway")

## Create a pathway assay from gene data with verbose output
#spamtp_obj <- createPathwayAssay(spamtp_obj, analyte_type = "genes", assay = "SPT", new_assay = "gene_pathway", v
```

createPathwayObject	<i>Store pathway scores in an alternative experiment</i>
---------------------	--

Description

This function computes pathway-level scores from analyte-level expression data and stores them in altExp for SingleCellExperiment-derived objects. Each score is the sum of matched expression divided by the square root of the matched feature count, then centred and scaled across pixels. Constant scores are zero; unmatched pathways are removed or stored as NA.

Usage

```
createPathwayObject(
  object,
  assay = NULL,
  slot = "logcounts",
  new.assay = "pathway",
  remove.nans = TRUE,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  pathway_index = NULL,
  duplicate_genes = c("error", "mean", "sum"),
  min_path_size = 1,
  max_path_size = Inf
)
```

Arguments

object	A SingleCellExperiment (including SpatialExperiment) with expression features indexed by gene identifiers or RaMP IDs.
assay	Primary or alternative experiment to score.
slot	Expression assay. If omitted for a Bioconductor container, prefers logcounts, normcounts, then counts in the selected experiment. See experimentAccess for the experiment/matrix distinction.
new.assay	Name for the pathway-level altExp() (default = "pathway"). Scores are stored in that experiment's pathwayScores expression assay.
remove.nans	Remove pathways without matched analytes (default TRUE).

database	Optional named list of database resources, normally created by <code>loadSpaMTPDatabase()</code> .
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see <code>loadSpaMTPDatabase()</code> .
database_local_dir	Optional staged SpaMTPdb resource directory.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with <code>gene_reference</code> or <code>gene_index</code> . "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through <code>SpaMTPdb : : spaMTPdbGeneReference</code> . Custom references require unique <code>hgnc_id</code> and <code>symbol</code> ; optional fields are <code>status</code> , <code>entrez_id</code> , <code>ensembl_gene_id</code> , <code>uniprot_ids</code> , <code>prev_symbol</code> , <code>alias_symbol</code> .
gene_index	Optional index built from the same source table by <code>buildGeneMappingIndex()</code> . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
pathway_index	A reusable index from <code>buildPathwayIndex()</code> . When supplied, its gene identities and memberships must match any explicit database or <code>gene_index</code> . Fingerprints prevent incompatible reuse.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.
min_path_size	Minimum measured unique members to score (default 1).
max_path_size	Maximum measured unique members to score (default Inf).

Details

Accepts gene identifiers directly or a harmonized RaMP assay. `min_path_size` and `max_path_size` are applied to measured unique members. `rowData` contains `database_size` (after identity reconciliation), `database_raw_size`, `measured_size`, `coverage_fraction`, `used_size`, `database_members`, `measured_members`, `used_members` and `excluded_conflict_records`. The complete coverage table, including excluded and unmeasured pathways, is retained in `metadata(altExp(x, new.assay))$pathway_mapping$coverage`. Scores summarize observed expression; they are not pathway activity tests.

Value

The input with pathwayScores in a new alternative experiment. Pathway identifiers are preserved.

Examples

```
utils::str(formals(createPathwayObject))
```

db_adduct_filter	<i>Filters the provided metabolomic database by polarity and adducts</i>
------------------	--

Description

Filters the provided metabolomic database by polarity and adducts

Usage

```
db_adduct_filter(db, adduct, polarity = "neg", verbose = TRUE)
```

Arguments

db	DataFrame of the current reference database.
adduct	Character string defining the adduct to be checked.
polarity	Character string defining the polarity of the adducts (default = "neg").
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Value

A filtered reference metabolomic database DataFrame

Examples

```
# HMDB_db <- load("data/HMDB_1_names.rds")  
# db_adduct_filter(HMDB_db,c("M+H"), polarity = "pos")
```

deconvolveSMILES	<i>Decompose explicit functional groups in SMILES strings</i>
------------------	---

Description

A lightweight, dependency-free SMILES graph parser identifies common metabolite functional groups and derives interpretable positive-ion, negative-ion, alkali-binding, and neutral-mass priors. The parser is intended for candidate prioritisation rather than pKa, proton-affinity, or quantum chemistry calculations. Stereochemistry is accepted but does not change the counts.

Usage

```
deconvolveSMILES(  
  smiles,  
  backend = c("auto", "native"),  
  strict = FALSE,  
  workers = getOption("SpaMTP.smiles_workers", 1L)  
)
```

Arguments

smiles	Character vector of SMILES strings.
backend	Currently "auto" and "native" both use SpaMTP's native parser. The argument reserves a stable extension point for optional chemistry toolkits.
strict	Stop when any SMILES cannot be parsed. When FALSE, invalid rows are returned with <code>structure_valid = FALSE</code> and missing features.
workers	Number of forked workers used for unique SMILES on platforms supporting <code>parallel::mclapply()</code> . The default is controlled by <code>options(SpaMTP.smiles_workers = 1)</code> .

Value

A data frame with functional-group counts, ion-mode scores, and a human-readable `structure_evidence` field.

Examples

```
utils::str(formals(deconvolveSMILES))
```

demsHeatmap

*Heatmap of Differentially Expressed Metabolites***Description**

Generates a heatmap of DEMs generated from edgeR analysis run using findAllDEMs(). This function uses pheatmap to plot data.

Usage

```
demsHeatmap(
  edgeR_output,
  n = 5,
  only.pos = FALSE,
  FDR.threshold = 0.05,
  logfc.threshold = 0.5,
  order.by = "FDR",
  scale = "row",
  color = (grDevices::colorRampPalette(c("navy", "white", "red")))(50),
  cluster_cols = FALSE,
  cluster_rows = TRUE,
  fontsize_row = 15,
  fontsize_col = 15,
  cutree_cols = 9,
  silent = TRUE,
  plot_annotatons_column = NULL,
  save_to_path = NULL,
  plot.save.width = 20,
  plot.save.height = 20,
  nlabels.to.show = NULL,
  annotation_colors = NULL
)
```

Arguments

- | | |
|-----------------|--|
| edgeR_output | A list containing outputs from edgeR analysis (from findAllDEMs()). This includes pseudo-bulked counts and DEMs. |
| n | A numeric integer that defines the number of UP and DOWN regulated peaks to plot (default = 25). |
| only.pos | Boolean indicating if only positive markers should be returned (default = FALSE). |
| FDR.threshold | Numeric value that defines the FDR threshold to use for defining most significant results (default = 0.05). |
| logfc.threshold | Numeric value that defines the logFC threshold to use for filtering significant results (default = 0.5). |

<code>order.by</code>	Ranking column: FDR, logFC, mean_auc or mean_cohen. Descriptive native marker results default to mean_auc; other inputs use FDR. FDR ranking is unavailable for descriptive marker results.
<code>scale</code>	A character string indicating if the values should be centered and scaled in either the row direction or the column direction, or none. Corresponding values are "row", "column" and "none"
<code>color</code>	A vector of colors used in heatmap (default = <code>grDevices::colorRampPalette(c("navy", "white", "red"))(50)</code>).
<code>cluster_cols</code>	Boolean value determining if columns should be clustered or hclust object (default = FALSE).
<code>cluster_rows</code>	Boolean value determining if rows should be clustered or hclust object (default = TRUE).
<code>fontsize_row</code>	A numeric value defining the fontsize of rownames (default = 15).
<code>fontsize_col</code>	A numeric value defining the fontsize of colnames (default = 15).
<code>cutree_cols</code>	A numeric value defining the number of clusters the columns are divided into, based on the hierarchical clustering(using cutree), if cols are not clustered, the argument is ignored (default = 9).
<code>silent</code>	Boolean value indicating if the plot should not be draw (default = TRUE).
<code>plot_annotatons_column</code>	Character string indicating the column name that contains the metabolite annotations to plot. Annotations = TRUE must be used in <code>findAllDEMs()</code> for edgeR output to include annotations. If <code>plot_annotatons_column</code> = NULL, m/z vaues will be plotted (default = NULL).
<code>save_to_path</code>	Character string defining the full filepath and name of the plot to be saved as.
<code>plot.save.width</code>	Integer value representing the width of the saved pdf plot (default = 20).
<code>plot.save.height</code>	Integer value representing the height of the saved pdf plot (default = 20).
<code>nlabels.to.show</code>	Numeric value defining the number of annotations to show per m/z (default = NULL).
<code>annotation_colors</code>	List for specifying <code>annotation_row</code> and <code>annotation_col</code> track colors manually. Check pheatmap R-Package documentation for details. If set to 'NA', default coloring will be used (default = NA).

Value

A heatmap plot of significantly differentially expressed metabolites defined in the edgeR ouput object.

Examples

```
utils::str(formals(demsHeatmap))

# demsHeatmap(DEMs)
```

densityMap*Export an interactive spatial detection-density viewer*

Description

The spatial kernel uses coordinates of nonzero pixels, not intensity weights. The mass-spectrum overlay uses an intensity-weighted density without replicating each mass according to its intensity. This exploratory viewer is not a spatial significance test. Its HTML requires internet access to load the existing Plotly and JavaScript KDE libraries.

Usage

```
densityMap(  
  object,  
  assay = "main",  
  slot = "counts",  
  folder = getwd(),  
  sampleId = NULL,  
  ...  
)
```

Arguments

object	A SpatialExperiment or aligned Cardinal experiment.
assay	Primary or alternative experiment.
slot	Non-negative intensity assay. See experimentAccess for the experiment/matrix distinction.
folder	Output directory.
sampleId	Sample ID; required for multiple samples.
...	Reserved; unused arguments cause an error.

Value

Invisibly, the path to mzs_density_map.html.

Examples

```
utils::str(formals(densityMap))
```

experimentAccess

Access SpaMTP data through Bioconductor containers

Description

SpaMTP uses public S4 accessors, not internal slots. A SpatialExperiment is a SingleCellExperiment and a SummarizedExperiment. The required container class is stated on each function's help page; spatial operations need coordinates, while paired modalities and reductions need SingleCellExperiment infrastructure. Raw, unaligned MSI spectra remain in Cardinal until binned or aligned to a common feature-by-pixel matrix.

Value

This is a documentation topic, not a callable function. The accessors described here return the selected data: assay() returns a matrix-like object, altExp() returns an alternative experiment, rowData() and colData() return DataFrame objects, and spatialCoords() returns a coordinate matrix. Replacement accessors update the corresponding data in the experiment. See each accessor's help page for its full return contract.

Experiments and expression assays

In analysis functions, assay selects the primary experiment ("main") or a named SingleCellExperiment::altExp(). The compatibility arguments slot, slots, SM.slot, ST.slot, SM_slot, ST_slot, mz.slot and pathway.slot select expression assay names within those experiments. They do not name S4 slots. Supply exact, non-empty character names, not numeric matrix positions. For example, assay = "transcriptome" and slot = "logcounts" read SummarizedExperiment::assay(SingleCellExperiment::altExp(x, "transcriptome"), "logcounts").

The expression assays within one experiment share feature and pixel axes; counts, normcounts, logcounts and scaled represent different value scales, not different modalities. Each alternative experiment has its own feature rows and the same paired pixel columns. Inspect assayNames() and altExpNames() before selecting data. No automatic alias maps data to logcounts or scale.data to scaled during matrix access.

For older analysis calls, NULL, "primary", "Spatial", "SPM", the primary experiment's mainExpName(), and its expression assay names can also select the primary experiment. An exact alternative-experiment name takes precedence over these aliases. Prefer "main" and explicit altExpNames() in new code. An expression name supplied as assay does not replace the separate slot selection. Constructors and explicit converters have their own documented naming arguments.

Metadata and spatial infrastructure

Use SummarizedExperiment::rowData() for feature annotations and SummarizedExperiment::colData() for pixel metadata. A primary experiment and each altExp() have separate rowData() and S4Vectors::metadata(). Parent pixel metadata are not automatically copied into an extracted alternative experiment. Use SingleCellExperiment::colLabels() for groups, reducedDim() for embeddings and colPair() for pixel-pair graphs. SpatialExperiment coordinates and images are accessed with spatialCoords(), imgData() and imgRaster().

Accessors have replacement forms. Standard column subsetting keeps paired alternative experiments and spatial coordinates aligned. Row subsetting changes only the selected experiment's feature space. SpaMTP does not require users to define additional S4 slots or inspect internal storage.

Optional Seurat conversion

Core analysis does not use Seurat. SeuratObject is an optional dependency for `seuratToSingleCellExperiment()`, `seuratToSpatialExperiment()` and `spatialExperimentToSeurat()`. Convert once at the boundary, then use the native Bioconductor workflow. Seurat assay names become experiment names; selected Seurat layers become expression assays as documented by the converter.

See Also

`asSpatialExperiment()`, `addTranscriptome()`, `normalizeSMDData()`, `scaleSMDData()`, `runMetabolicPCA()`

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = rbind(a = 1:4, b = 4:1)),
  spatialCoords = cbind(x = 1:4, y = 0),
  colData = S4Vectors::DataFrame(row.names = paste0("p", 1:4)))
rna <- matrix(c(1, 3, 2, 4), nrow = 1,
  dimnames = list("gene1", colnames(x)))
x <- addTranscriptome(x, rna)
x <- normalizeSMDData(x, "LogNormalize", assay = "transcriptome",
  slot = "counts", verbose = FALSE)
rna <- SingleCellExperiment::altExp(x, "transcriptome")
SummarizedExperiment::assayNames(rna)
SummarizedExperiment::assay(rna, "logcounts")
SummarizedExperiment::rowData(rna)$feature_label <- "example gene"
SingleCellExperiment::altExp(x, "transcriptome") <- rna
SummarizedExperiment::colData(x)$region <- c("A", "A", "B", "B")
selected <- x[, x$region == "A"]
stopifnot(identical(colnames(SingleCellExperiment::altExp(selected,
  "transcriptome")), colnames(selected)))
```

findAllDEMs

Finds differentially expressed m/z values/metabolites between all comparison groups.

Description

Finds differentially expressed m/z values/metabolites between all comparison groups.

Usage

```
findAllDEMs(
  data,
  ident,
  n = 3,
  logFC_threshold = 1.2,
  DE_output_dir = NULL,
  run_name = "findAllDEMs",
  annotation.column = NULL,
  assay = "Spatial",
  slot = "counts",
  return.individual = FALSE,
  verbose = TRUE,
  seed = 1234,
  method = c("technical", "markers", "replicate"),
  replicate = NULL,
  contrasts = list(),
  min_region_size = 3,
  spatial_blocks = 0
)
```

Arguments

<code>data</code>	A SingleCellExperiment, including SpatialExperiment, for markers or replicate mode. Technical pooling also accepts a plain SummarizedExperiment.
<code>ident</code>	Column name in <code>SummarizedExperiment::colData(data)</code> defining the groups to compare. Technical mode requires this column to be a factor.
<code>n</code>	An integer that defines the number of pseudo-replicates (pools) per sample (default = 3).
<code>logFC_threshold</code>	Technical-mode fold-change threshold, passed as <code>log2(logFC_threshold)</code> to <code>limma::treat()</code> (default = 1.2). Not used to filter marker or biological-replicate results.
<code>DE_output_dir</code>	A character string defining the directory path for all output files to be stored. This path must a new directory. Else, set to NULL as default.
<code>run_name</code>	A character string defining the title of this DE analysis that will be used when saving DEMs to .csv file (default = 'findAllDEMs').
<code>annotation.column</code>	Optional column in the selected experiment's <code>rowData()</code> , such as "all_IsomerNames" from <code>annotateSM()</code> (default NULL).
<code>assay</code>	Primary ("main") or alternative experiment name. The default "Spatial" is a compatibility alias for the primary experiment.
<code>slot</code>	Expression assay name within the selected experiment (default = "counts"). Choose the value scale for the analysis: technical pooling mode requires finite non-negative unlogged values; marker and replicate modes use the supplied values directly, commonly "logcounts". See experimentAccess for the experiment/matrix distinction.

return.individual	In technical mode, return one result per group instead of a merged result. Unsupported in marker/replicate modes.
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
seed	Numeric value used to set the seed for reproducible randomisation (default = 1234).
method	markers uses scan effect-size summaries without pixel P values; replicate fits configured contrasts on biological-replicate means. The historical technical mode is retained for compatibility, but random pools are technical subdivisions and cannot establish biological replication. The compatibility default is "technical"; choose "markers" or "replicate" explicitly for the corresponding native analysis.
replicate	Complete colData field identifying biological replicates.
contrasts	Named list of numerator, denominator and paired settings, required for replicate inference. At least three replicates per arm or three complete pairs are required.
min_region_size	Minimum labelled observations in a marker group.
spatial_blocks	Number of spatial blocks per sample for a leave-one-block sensitivity analysis of marker effects, or zero to disable. Blocks use coordinates only. They are not biological replicates or confidence intervals.

Value

A result list, not a modified experiment. Technical mode returns pooled values in counts and differential results in DEMs (or a list per group when `return.individual = TRUE`). Marker mode returns DEMs, scores, expression, group eligibility and provenance. Replicate mode returns `group_means` and named tests with status and, when eligible, limma tables and the biological-replicate design. Marker mode may return `status = "skipped"` with a reason when too few groups are eligible.

Examples

```
utils::str(formals(findAllDEMs))
```

findCorrelatedFeatures

Find features correlated with a metabolite, gene, or pixel group

Description

Computes Pearson correlations across matched pixels. Genes are read from altExp without assigning artificial m/z values. Constant features return NA.

Usage

```
findCorrelatedFeatures(
  data,
  mz = NULL,
  gene = NULL,
  ident = NULL,
  SM.assay = "main",
  ST.assay = NULL,
  SM.slot = "counts",
  ST.slot = "counts",
  nfeatures = 10,
  covariates = NULL,
  features = NULL
)
```

Arguments

data	A SpatialExperiment or aligned MSIImagingExperiment.
mz	Numeric m/z or exact primary feature name.
gene	Exact feature name in ST.assay.
ident	A colData column defining groups. Supply exactly one of mz, gene and ident. Groups are compared with binary membership vectors.
SM.assay	MSI experiment name; main selects the primary experiment. Legacy SPM and Spatial also select the primary experiment.
ST.assay	Alternative experiment name, or NULL for metabolites only.
SM.slot	Assay containing metabolite values. See experimentAccess for the experiment/matrix distinction.
ST.slot	Assay containing transcriptomic values. See experimentAccess for the experiment/matrix distinction.
nfeatures	Maximum results per reference; NULL returns all features.
covariates	Optional colData fields to regress from both features before correlation, for example sample and annotated region. Raw correlations are also returned. This is a descriptive conditional association, not a causal or spatial significance test.
features	Optional named list of exact target IDs, with entries metabolite and/or gene. Selection does not change the reference vector.

Value

A data frame with features, correlation, modality, mz, rank and, for multiple reference groups, ident.

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = rbind(a = 1:4, b = 4:1)),
  rowData = S4Vectors::DataFrame(mz = c(100, 200)),
  spatialCoords = cbind(x = 1:4, y = 0),
```

```
colData = S4Vectors::DataFrame(row.names = paste0("p", 1:4))  
findCorrelatedFeatures(x, mz = 100)
```

```
findDuplicateAnnotations
```

Finds if any metabolite is duplicated across multiple m/z values.

Description

Finds if any metabolite is duplicated across multiple m/z values.

Usage

```
findDuplicateAnnotations(data, assay = "main")
```

Arguments

data	Bioconductor experiment containing annotated m/z values.
assay	Character string defining the primary or alternative experiment that contains the annotated metadata corresponding to the m/z values (default = "Spatial").

Value

Vector of character strings describing metabolites that are assigned to multiple m/z values

Examples

```
utils::str(formals(findDuplicateAnnotations))
```

```
findNearestMZ
```

Finds the nearest m/z peak to a given value in a SpaMTP Object

Description

Finds the nearest m/z peak to a given value in a SpaMTP Object

Usage

```
findNearestMZ(data, target_mz, assay = NULL)
```

Arguments

data	A SummarizedExperiment, including SingleCellExperiment or SpatialExperiment, with feature m/z values in rowData() or feature names.
target_mz	Numeric value defining the target m/z peak
assay	Primary ("main" or NULL) or alternative experiment name. The mass lookup uses the selected experiment's rowData(), not an expression layer; see experimentAccess .

Value

The exact character feature identifier for the nearest measured m/z, not a numeric mass. Ties select the first feature in the stored order.

Examples

```
utils::str(formals(findNearestMZ))
# findNearestMZ(spe, target_mz = 400.01)
```

findRegionalPathways *Regional Pathway Enrichment*

Description

This the function used to compute the gene/metabolites set enrichment for multi-omics spatial data

Usage

```
findRegionalPathways(
  SpaMTP,
  ident,
  DE.list = NULL,
  analyte_types = c("genes", "metabolites"),
  SM_assay = "main",
  ST_assay = "transcriptome",
  SM_slot = "counts",
  ST_slot = "counts",
  min_path_size = 5,
  max_path_size = 500,
  pval_cutoff_mets = 0.05,
  pval_cutoff_genes = 0.05,
  annotation_score_threshold = 0.05,
  annotation_source = c("current", "auto", "legacy"),
  verbose = TRUE,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  pathway_index = NULL,
  ranks = NULL,
  rank_scale = c("rms", "none"),
```

```

    nPermSimple = 1000
)

```

Arguments

SpaMTP	A Bioconductor experiment contains spatial metabolomics(SM)/transcriptomics(ST) data or both, if contains SM data, it should be annotated via SpaMTP::annotateSM function.
ident	A character scalar specifying the region column in cell metadata.
DE.list	A list consisting of differential expression data.frames for each input modality. Within each data.frame column names MUST include 'cluster', 'gene', ('avg_log2FC' or 'logFC') and ('p_val_adj' or 'FDR').
analyte_types	Vector of character strings defining which analyte types to use. Options can be c("genes"), c("metabolites") or both (default = c("genes", "metabolites")).
SM_assay	Primary MSI experiment name (default = "main").
ST_assay	Paired transcriptome altExp name (default = "transcriptome").
SM_slot	Expression assay name within the metabolomics experiment selected by SM_assay (default = "counts"). See experimentAccess for the experiment/matrix distinction.
ST_slot	Expression assay name within the transcriptomics experiment selected by ST_assay (default = "counts"). See experimentAccess for the experiment/matrix distinction.
min_path_size	The min number of metabolites in a specific pathway (default = 5).
max_path_size	The max number of metabolites in a specific pathway (default = 500).
pval_cutoff_mets	Adjusted p-value cutoff used when constructing metabolite ranks. Set to 1 to include annotated metabolites regardless of DE significance (default = 0.05).
pval_cutoff_genes	A numerical value defining the adjusted p value cutoff for significant differentially expressed genes. If NULL cutoff = 0.05 (default = 0.05).
annotation_score_threshold	Minimum indexed annotation score used for pathway mapping. It can be changed without re-running annotateSM() when the object was annotated with min_score = 0 (default = 0.05).
annotation_source	Metabolite annotation provenance. The default, "current", requires scored RaMP IDs from the indexed annotation pipeline. "auto" permits a warned fall-back to a legacy stored db_3, while "legacy" explicitly requests that compatibility path.
verbose	Boolean indicating whether to show informative messages. If FALSE these messages will be suppressed (default = TRUE).
database	Optional named list of database resources, normally created by loadSpaMTPDatabase() .
database_version	SpaMTPdb/RaMP version used for pathway lookup.

database_source	Database source; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spaMTPdbGeneRef . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
pathway_index	A reusable index from buildPathwayIndex() . When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
ranks	Alternative to DE.list: named genes/metabolites matrices with features in rows and regions in columns. Uses complete, unfiltered ranks. Metabolite rows must be explicit RaMP compound IDs. Map expression with createPathwayAssay before calculating identity-level effects.
rank_scale	RMS-scale each modality's rank vector before concatenating, or none. Pathways can still be dominated by the more densely measured modality.
nPermSimple	Preliminary permutations for the ranks interface.

Details

Gene mapping uses the same identity index as [fishersPathwayAnalysis\(\)](#). Multiple RaMP records of a gene have their memberships united. If multiple input features map to that gene with different differential statistics in one cluster, resolve the duplicate features before differential analysis; the function does not select an arbitrary statistic.

Value

A data frame of regional pathway enrichment results, not a modified experiment. The complete-ranks interface also attaches the ranks, pathway coverage, mapping and multiple-testing provenance as attributes.

Examples

```
utils::str(formals(findRegionalPathways))
```

```
findSpatiallyVariableMetabolites
```

Find spatially variable metabolites using Moran's I

Description

Calls `ape::Moran.I` with inverse squared Euclidean distance weights and zero diagonal, row-standardized by `ape`. Samples are analysed separately. Constant features return NA.

Usage

```
findSpatiallyVariableMetabolites(  
  object,  
  assay = "main",  
  slot = "counts",  
  image = NULL,  
  nfeatures = 2000,  
  max_spots = 5000,  
  seed = 1,  
  verbose = TRUE,  
  sampleId = NULL  
)
```

Arguments

<code>object</code>	A <code>SpatialExperiment</code> or aligned <code>MSImagingExperiment</code> .
<code>assay</code>	Primary ("main") or alternative experiment to analyse.
<code>slot</code>	Assay containing expression values. See experimentAccess for the experiment/matrix distinction.
<code>image</code>	Retained for compatibility; coordinates use <code>spatialCoords</code> .
<code>nfeatures</code>	Maximum variable features selected.
<code>max_spots</code>	Maximum pixels for the dense distance matrix; NULL uses all.
<code>seed</code>	Random seed for sampling; caller RNG state is restored.
<code>verbose</code>	Show progress messages.
<code>sampleId</code>	A <code>sample_id</code> in <code>colData</code> . Required for multiple samples.

Value

A `SpatialExperiment` with statistics, BH-adjusted p-values, flags and ranks in `rowData` of the selected experiment. Settings and sampled pixel IDs are stored in `metadata(object)$moransi`.

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = rbind(a = 1:6, b = c(1, 3, 5, 2, 4, 6))),
  rowData = S4Vectors::DataFrame(mz = c(100, 200)),
  spatialCoords = cbind(x = 1:6, y = 0),
  colData = S4Vectors::DataFrame(row.names = paste0("p", 1:6)))
x <- findSpatiallyVariableMetabolites(x)
getSpatiallyVariableMetabolites(x)
```

fishersPathwayAnalysis

Fisher Exact Tests for Pathway Enrichment

Description

Fisher Exact Tests for Pathway Enrichment

Usage

```
fishersPathwayAnalysis(
  Analyte,
  max_path_size = 500,
  min_path_size = 5,
  alternative = "greater",
  pathway_all_info = FALSE,
  pval_cutoff = NULL,
  verbose = TRUE,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  universe = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  pathway_index = NULL,
  ...
)
```

Arguments

Analyte	Foreground analytes as a named list with elements genes, metabolites and/or mzs. Identifiers are matched case-insensitively to source IDs (e.g. <code>gene_symbol:TP53</code> ,
---------	---

	hmdb:HMDB0000122), known RaMP IDs, then common names for otherwise unmatched queries. Duplicate mappings to the same RaMP ID count once. m/z values may be numeric or numeric strings, optionally prefixed with mz- or mz_.
max_path_size	Maximum pathway size within the analysis universe (inclusive; default 500).
min_path_size	Minimum pathway size within the analysis universe (inclusive; positive integer, default 5).
alternative	Fisher test alternative: "greater", "less" or "two.sided".
pathway_all_info	Include names, input IDs and m/z adduct information for foreground members of each pathway. Does not change the tests.
pval_cutoff	Optional BH-adjusted p-value (FDR) cutoff in $[0, 1]$. Applied only after correcting over all eligible pathways.
verbose	Whether to print progress messages. Mapping warnings are always emitted, including when verbose = FALSE.
database	Optional named resource list from loadSpaMTPDatabase() . Identifier analysis requires source_df, analyte, analytehaspathway and pathway; m/z analysis additionally uses chem_props unless db or index is supplied through ...
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
universe	Optional measured background, in the same named-list format as Analyte. Supply all analytes eligible for foreground selection, not just significant hits. Foreground and universe must specify the same biological modalities (genes and/or compounds; mzs and metabolites both denote compounds). All mapped foreground IDs must belong to the mapped universe, otherwise an error is raised. Mapped background IDs without any pathway membership still contribute to the universe size. When NULL, use all pathway-linked IDs of the specified modalities in the database, before pathway-size filtering; mapped foreground IDs without pathway membership are then excluded with a warning.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb::spaMTPdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.

gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
pathway_index	A reusable index from buildPathwayIndex(). When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
...	Arguments passed to the indexed annotateTable() pipeline for m/z inputs, such as ppm_error, adducts, db or index.

Details

With universe size U , foreground size K , pathway size M and overlap a , the test table is `matrix(c(a, M - a, K - a, U - K - M + a), nrow = 2)`. Memberships and identifiers are deduplicated by RaMP ID. Pathway sizes are computed after intersecting with the universe; size filtering never changes that universe. All pathways passing the size limits are tested, including zero-overlap pathways. BH correction uses this complete family, before any output cutoff. Missing display metadata does not remove a test.

Unmapped input values are excluded with warnings and recorded in the enrichment attribute. An empty mapped universe is an error. An empty foreground gives p-values of 1 for all eligible pathways. If no pathways pass the size limits, a zero-row data frame is returned.

For a targeted panel, provide the measured panel as universe; the default database background is not a substitute for the detection background. m/z foreground and background values are annotated together with identical settings. All mapped candidate RaMP IDs are retained; ambiguous mass matches are not confirmed compound identifications. In a combined analysis, genes and compounds are counted as individual RaMP IDs in a single pooled test.

Value

A data frame sorted by `p_val`, retaining the columns `pathway_name`, `pathway_id`, `type`, `pathwayCategory`, `p_val`, `fdr`, `ratio`, `analytes_in_pathways` (overlap) and `total_in_pathways` (size within the universe). Also includes `pathwayRampId`, `foreground_analytes_number` and `background_analytes_number`. `pathway_all_info = TRUE` adds member information. The enrichment attribute records mapped foreground and universe IDs, unmapped inputs, excluded foreground IDs, universe source, alternative, size limits and the number of tests before output filtering. Its `gene_mapping` entry contains per-input mapping status, gene groups, excluded conflicting RaMP records and versioned reference provenance.

Examples

```
utils::str(formals(fishersPathwayAnalysis))
# For a targeted gene panel:
# fishersPathwayAnalysis(
#   Analyte = list(genes = paste0("gene_symbol:", significant_genes)),
#   universe = list(genes = paste0("gene_symbol:", measured_panel_genes))
# )
```

formula_filter	<i>Filters reference Database to only select natural elements</i>
----------------	---

Description

Filters reference Database to only select natural elements

Usage

```
formula_filter(df, elements = NULL)
```

Arguments

df	DataFrame of the reference database.
elements	Vector of character strings of elements to include (default = c("H", "C", "N", "O", "S", "Cl", "Br", "F", "Na", "P", "I", "Si")).

Value

A refined DataFrame which only includes annotations containing the specified elements

Examples

```
## Get filtered DB by adduct
# db_1 <- db_adduct_filter(db, test_add_pos, polarity = "pos")

## Refine to only include natural elements
# db_2 <- formula_filter(db_1)
```

get_analytes_db	<i>Helper function for building a pathway db based on detected</i>
-----------------	--

Description

Helper function for building a pathway db based on detected

Usage

```
get_analytes_db(input_id, analytehaspathway, chem_props, pathway)
```

Arguments

input_id	Vector of characters defining the detected .
analytehaspathway	A dataframe containing RAMP_pathway ID's.
chem_props	A database containing the chemical properties and metadata of each RAMP_DB analyte.
pathway	A dataframe containing RAMP_DB pathways and their relative metadata

Value

A analyte database containing corresponding pathways associated with each detected metabolite

Examples

```
#HELPER FUNCTION
```

getKmeanClusters	<i>Perform K-means clustering on a specified reduction</i>
------------------	--

Description

Runs K-means on an embedding read through `SingleCellExperiment::reducedDim()` and writes pixel cluster labels to `SummarizedExperiment::colData()`.

Usage

```
getKmeanClusters(
  data,
  reduction = "SpatialPCA",
  cluster.name = "spatial_clusters",
  clusters = 8,
  iter.max = 10,
  nstart = 1,
  algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"),
  trace = FALSE,
  seed = 888
)
```

Arguments

<code>data</code>	A <code>SingleCellExperiment</code> , including <code>SpatialExperiment</code> , containing an embedding such as the result of <code>runSpatialGraphPCA()</code> .
<code>reduction</code>	Name in <code>SingleCellExperiment::reducedDimNames(data)</code> (default = "SpatialPCA").
<code>cluster.name</code>	Column name in <code>SummarizedExperiment::colData(data)</code> for cluster labels (default = "spatial_clusters").
<code>clusters</code>	Integer defining the number of clusters to form (default = 8).
<code>iter.max</code>	Integer defining the maximum number of iterations allowed (default = 10).
<code>nstart</code>	Integer stating the number of random sets to choose (default = 1).
<code>algorithm</code>	Character string defining the K-means algorithm to use. One of "Hartigan-Wong", "Lloyd", "Forgy", or "MacQueen" (default = "Hartigan-Wong").
<code>trace</code>	Logical boolean indicating whether to produce tracing information on the progress of the algorithm (default = FALSE).
<code>seed</code>	Integer of the random seed to use for reproducibility (default = 888).

Value

The input container with cluster assignments in `SummarizedExperiment::colData(data)[[cluster.name]]`.

Examples

```
utils::str(formals(getKmeanClusters))
```

getMetaspace	<i>Core METASPACE Data Retrieval</i>
--------------	--------------------------------------

Description

The main user-facing function to download annotations and ion images for a METASPACE dataset.

Usage

```
getMetaspace(
  dataset_id,
  fdr = 0.1,
  database = c("HMDB", "v4"),
  include_images = TRUE,
  api_key = NULL,
  isotope_idx = 1,
  relative = TRUE
)
```

Arguments

<code>dataset_id</code>	Character string of the METASPACE dataset ID.
<code>fdr</code>	Numeric, the FDR level threshold (default = 0.1).
<code>database</code>	Character vector, specifying the database name and version (default = <code>c("HMDB", "v4")</code>).
<code>include_images</code>	Logical, if TRUE, downloads all ion image matrices (default = TRUE).
<code>api_key</code>	Optional character string, your METASPACE API key (default = NULL).
<code>isotope_idx</code>	Integer, index of the isotope image to download for all annotations (1 for main peak) (default = 1).
<code>relative</code>	Logical, if TRUE, normalizes pixel intensities (0-1 range) (default = TRUE).

Value

A list with two elements: `annotations` (data.frame) and `images` (list of 2D matrices).

Examples

```
utils::str(formals(getMetaspace))
# getMetaspace(dataset_id = "2020-12-07_03h16m14s", fdr = 0.1, database= c("HMDB", "v4"), relative = TRUE)
```

getMZMetadata	<i>Gets values from a single metadata column for a respective m/z value.</i>
---------------	--

Description

Gets values from a single metadata column for a respective m/z value.

Usage

```
getMZMetadata(  
  obj,  
  mz,  
  assay = "main",  
  metadata.column = "all_IsomerNames",  
  separate = TRUE  
)
```

Arguments

obj	Bioconductor experiment containing annotated m/z values.
mz	Character string specifying the m/z value to return the metadata for.
assay	Character string defining the primary or alternative experiment that contains the annotated metadata corresponding to the m/z values (default = "Spatial").
metadata.column	Character string corresponding to the assay feature metadata column to extract (default = "all_IsomerNames").
separate	Boolean indicating whether to separate the metadata string and return a vector. Note, if TRUE the metadata column should contain values separate by ";" (default = TRUE).

Value

Vector of character strings containing the metadata for a specific mz value.

Examples

```
utils::str(formals(getMZMetadata))  
# getMZMetadata(SpaMTP, mz = "mz-100", metadata.column = "all_IsomerNames")  
  
##### Example for getting metabolite annotation IDs for a m/z value  
# getMZMetadata(SpaMTP, mz = "mz-100", metadata.column = "all_Isomers")
```

getRefinedAnnotations *Refines and reduces m/z annotations*

Description

Used to subset dataset to only include annotations that have n number of entries. For example some peaks can have multiple annotations. Peaks which have above n number of annotations assigned will be removed from Bioconductor experiment.

Usage

```
getRefinedAnnotations(obj, assay = "main", n = 1)
```

Arguments

obj	A SummarizedExperiment, including SingleCellExperiment or SpatialExperiment. The selected experiment must have annotations in rowData().
assay	Primary ("main") or alternative experiment whose rowData() contains the annotations (default = "main").
n	Integer defining the number of entries an annotation can have assigned. Any higher counts will be removed (default = 1).

Value

The input container with the selected experiment restricted to features having at most n annotations. Pixel columns are unchanged.

Examples

```
utils::str(formals(getRefinedAnnotations))  
# HMDB_db <- load("data/HMDB_1_names.rds")
```

getSpatiallyVariableMetabolites
Get the top spatially variable metabolites

Description

Get the top spatially variable metabolites

Usage

```
getSpatiallyVariableMetabolites(object, assay = "main", n = 10)
```

Arguments

object	A SpatialExperiment returned by findSpatiallyVariableMetabolites.
assay	Primary or alternative experiment containing results.
n	Maximum number of feature names.

Value

Feature names in Moran's I rank order, excluding untested features.

getSquareCoordinates	<i>This function computes the coordinates of a square's four corners based on a given center point and width.</i>
----------------------	---

Description

This function computes the coordinates of a square's four corners based on a given center point and width.

Usage

```
getSquareCoordinates(center_x, center_y, width, name)
```

Arguments

center_x	Numeric value defining the x-coordinate of the square's center.
center_y	Numeric value defining the y-coordinate of the square's center.
width	Numeric value indicating the width of the square.
name	Character string specifying the label assigned to the square for identification.

Value

A data frame with columns Selection, X, and Y representing the square's name and corner coordinates.

Examples

```
utils::str(formals(getSquareCoordinates))
getSquareCoordinates(center_x = 5, center_y = 5, width = 4, name = "MySquare")
```

imageMZAnnotationPlot *Plot annotated metabolites without an optical image*

Description

All matching peaks are summed per metabolite. A peak shared by overlapping mass-tolerance windows is counted once. No temporary metadata is written.

Usage

```
imageMZAnnotationPlot(  
  object,  
  metabolites,  
  plusminus = NULL,  
  fov = NULL,  
  cols = c("lightgrey", "firebrick1"),  
  size = 0.5,  
  min.cutoff = NA,  
  max.cutoff = NA,  
  split.by = NULL,  
  alpha = 1,  
  border.color = "white",  
  border.size = 0,  
  dark.background = TRUE,  
  crop = FALSE,  
  cells = NULL,  
  scale = c("feature", "all", "none"),  
  axes = FALSE,  
  combine = TRUE,  
  coord.fixed = TRUE,  
  assay = "main",  
  slot = "counts",  
  column.name = "all_IsomerNames",  
  plot.exact = TRUE,  
  plot.pixel = FALSE,  
  verbose = TRUE,  
  ...  
)
```

Arguments

object	A SpatialExperiment or aligned Cardinal MSImagingExperiment.
metabolites	Metabolite names to search in feature annotations.
plusminus	Optional non-negative absolute m/z tolerance.
fov	Optional sample IDs from colData(object)\$sample_id.
cols	Gradient colours.

<code>size</code>	Point size.
<code>min.cutoff, max.cutoff</code>	One cutoff or one per query; numeric or quantile strings such as "q10". NA uses the observed range.
<code>split.by</code>	Optional colData column for separate panels within samples.
<code>alpha</code>	Point opacity, or a two-value range mapped to expression.
<code>border.color, border.size</code>	Point border colour and width.
<code>dark.background</code>	Use a black plot background.
<code>crop</code>	Crop to the selected pixels.
<code>cells</code>	Optional pixel IDs to display.
<code>scale</code>	Colour limits: "feature" across samples, "all" across all queries, or "none" separately for each panel.
<code>axes</code>	Show axes.
<code>combine</code>	Combine panels; FALSE returns a list of ggplots.
<code>coord.fixed</code>	Use a fixed coordinate aspect ratio.
<code>assay</code>	Primary experiment ("main") or altExp name.
<code>slot</code>	Expression assay name, e.g. "counts" or "logcounts". See experimentAccess for the experiment/matrix distinction.
<code>column.name</code>	Annotation column in rowData.
<code>plot.exact</code>	Match complete semicolon-delimited names (case-insensitive).
<code>plot.pixel</code>	Draw square rather than circular point symbols.
<code>verbose</code>	Retained for call compatibility.
<code>...</code>	Former Seurat segmentation, molecule and blending arguments are unsupported and cause an error; convert and plot these separately.

Value

A ggplot or list of panels.

Examples

```
utils::str(formals(imageMZAnnotationPlot))
```

imageMZPlot

*Plot MSI features from a Bioconductor spatial container***Description**

These functions read the primary experiment or a named altExp without modifying the object. Numeric m/z queries select the nearest measured mass; character feature IDs select exact rows. A mass tolerance sums all peaks within that distance of the selected measured mass. Overlapping annotation windows count each peak once. Multiple samples are plotted separately.

Usage

```
imageMZPlot(
  object,
  mzs,
  plusminus = NULL,
  fov = NULL,
  cols = c("lightgrey", "firebrick1"),
  size = 0.5,
  min.cutoff = NA,
  max.cutoff = NA,
  split.by = NULL,
  alpha = 1,
  border.color = "white",
  border.size = 0,
  dark.background = TRUE,
  crop = FALSE,
  cells = NULL,
  scale = c("feature", "all", "none"),
  axes = FALSE,
  combine = TRUE,
  coord.fixed = TRUE,
  assay = "main",
  slot = "counts",
  plot.pixel = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

object	A SpatialExperiment or aligned Cardinal MSImagingExperiment.
mzs	Numeric m/z values or character feature IDs.
plusminus	Optional non-negative absolute m/z tolerance.
fov	Optional sample IDs from colData(object)\$sample_id.
cols	Gradient colours.

<code>size</code>	Point size.
<code>min.cutoff, max.cutoff</code>	One cutoff or one per query; numeric or quantile strings such as "q10". NA uses the observed range.
<code>split.by</code>	Optional colData column for separate panels within samples.
<code>alpha</code>	Point opacity, or a two-value range mapped to expression.
<code>border.color, border.size</code>	Point border colour and width.
<code>dark.background</code>	Use a black plot background.
<code>crop</code>	Crop to the selected pixels.
<code>cells</code>	Optional pixel IDs to display.
<code>scale</code>	Colour limits: "feature" across samples, "all" across all queries, or "none" separately for each panel.
<code>axes</code>	Show axes.
<code>combine</code>	Combine panels; FALSE returns a list of ggplots.
<code>coord.fixed</code>	Use a fixed coordinate aspect ratio.
<code>assay</code>	Primary experiment ("main") or altExp name.
<code>slot</code>	Expression assay name, e.g. "counts" or "logcounts". See experimentAccess for the experiment/matrix distinction.
<code>plot.pixel</code>	Draw square rather than circular point symbols.
<code>verbose</code>	Retained for call compatibility.
<code>...</code>	Former Seurat segmentation, molecule and blending arguments are unsupported and cause an error; convert and plot these separately.

Value

A ggplot, or a list of panels when combine is FALSE.

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = rbind(a = 1:4, b = 4:1)),
  rowData = S4Vectors::DataFrame(mz = c(100, 200)),
  colData = S4Vectors::DataFrame(row.names = paste0("p", 1:4)),
  spatialCoords = cbind(x = 1:4, y = c(0, 1, 0, 1)))
imageMZPlot(x, mzs = 100)
```

`interactiveSpatialPlot`*Interactively inspect mass windows in a spatial experiment*

Description

The app sums every measured feature within the selected mass window without changing the input assays or annotations. The mass-axis controls use rowData masses, never mean intensities. No browser is launched by this function.

Usage

```
interactiveSpatialPlot(  
  obj,  
  assay = "main",  
  slot = "counts",  
  image = NULL,  
  sampleId = NULL  
)
```

Arguments

<code>obj</code>	A SpatialExperiment or aligned Cardinal experiment.
<code>assay</code>	Primary or alternative experiment.
<code>slot</code>	Expression assay. See experimentAccess for the experiment/matrix distinction.
<code>image</code>	Optional image ID from imgData.
<code>sampleId</code>	Optional sample ID. NULL displays separate sample panels.

Value

A Shiny application object; use shiny::runApp to launch.

Examples

```
utils::str(formals(interactiveSpatialPlot))
```

kNeighborsGraph

Construct a *k*-Nearest Neighbour Graph from Spatial Coordinates.

Description

Builds a sparse adjacency matrix representing the *k*-nearest neighbour relationships between spots or cells, using Euclidean distance between spatial coordinates.

Usage

```
kNeighborsGraph(
  location,
  n_neighbors,
  platform,
  include_self = FALSE,
  alg = "kd_tree"
)
```

Arguments

location	A data frame or matrix with columns <i>x</i> and <i>y</i> representing spatial coordinates, and rownames match barcode/cell names.
n_neighbors	Integer value defining the number of neighbours to consider.
platform	Character string being either "Visium" or "ST", specifying the platform of the dataset being analysed. Determines how distance thresholds and adjacency are calculated whereby "Visium" data is handled slightly differently to compensate for the hexagon shape of the spots.
include_self	Boolean logical value stating whether to include self-connections (default = FALSE).
alg	Character string specifying the algorithm to use for nearest neighbour search. Passed to <code>FNN::get.knn()</code> (default = "kd_tree").

Value

A sparse adjacency matrix (`dgCMatrix`) representing the *k*-NN graph.

Examples

```
utils::str(formals(kNeighborsGraph))
### HELPER FUNCTION
```

labels_to_show	<i>Filters the annotation list to only include the first n number of annotations per m/z</i>
----------------	--

Description

Filters the annotation list to only include the first n number of annotations per m/z

Usage

```
labels_to_show(annotation_column, n = 3)
```

Arguments

annotation_column	Vector from the rowData() column containing m/z annotations.
n	Numeric value defining the number of annotations to keep (default = 3).

Value

Vector containing the first n number of annotations

loadMetaspace	<i>Load METASPACE Data as a SpatialExperiment</i>
---------------	---

Description

Downloads METASPACE data, converts it to a feature matrix, and packages it in a Bioconductor SpatialExperiment. Use explicit conversion for other containers. Coordinates are stored in spatialCoords() without redundant copies in colData(); dataset and acquisition metadata remain in colData().

Usage

```
loadMetaspace(
  dataset_id,
  fdr = 0.1,
  database = c("HMDB", "v4"),
  api_key = NULL,
  relative = FALSE,
  transform = FALSE,
  verbose = TRUE,
  returnType = "SpatialExperiment"
)
```

Arguments

dataset_id	Character string of the METASPACE dataset ID.
fdr	Numeric, the FDR level threshold (default = 0.1).
database	Character vector, specifying the database name and version (default = c("HMDB", "v4")).
api_key	Optional character string, your METASPACE API key (default = NULL).
relative	Logical, if TRUE, normalizes pixel intensities (0-1 range) (default = FALSE).
transform	Boolean specifying whether to transform/flip the image (default = FALSE).
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
returnType	Output container; only "SpatialExperiment" is supported.

Value

A SpatialExperiment.

Examples

```
utils::str(formals(loadMetaspace))
# loadMetaspace(dataset_id = "2020-12-07_03h16m14s", fdr = 0.1, database= c("HMDB", "v4"), relative = TRUE)
```

loadSM

Load imzML data into a Bioconductor-native spatial container

Description

Raw spectra remain in Cardinal's file-backed MSImagingArrays, without asking the reader to infer a shared mass axis. Once a resolution is supplied and Cardinal has produced a common binned feature axis, the default return value is a SpatialExperiment. Seurat conversion is explicit and optional.

Usage

```
loadSM(
  file,
  mass.range = NULL,
  resolution = NA,
  units = "ppm",
  verbose = TRUE,
  assay = "Spatial",
  multi.run = FALSE,
  returnType = c("auto", "Cardinal", "SpatialExperiment")
)
```

Arguments

<code>file</code>	Character string defining the directory path of the file and the file name.
<code>mass.range</code>	Optional two-element mass range for binning. Requires resolution; raw spectral filtering should be performed in Cardinal.
<code>resolution</code>	Numeric value defining the the accuracy to which the m/z values will be binned after reading. This value can be in either "ppm" or "mz" depending on the units type specified (default = 10).
<code>units</code>	Character string defining the resolution value unit type, either c("ppm", "mz") (default = "ppm")
<code>verbose</code>	Boolean indicating whether to show informative processing messages. If TRUE the message will be show, else the message will be suppressed (default = TRUE)
<code>assay</code>	Label stored in mainExpName(); intensities are in the counts assay.
<code>multi.run</code>	Retained for compatibility; Cardinal run IDs are preserved.
<code>returnType</code>	Output container. "auto" returns Cardinal for raw data and SpatialExperiment after binning. SpatialExperiment output requires resolution.

Value

A Cardinal MSImagingArrays/MSImagingExperiment, or a SpatialExperiment.

Examples

```
utils::str(formals(loadSM))
# data <- loadSM(name = "run1", folder = "/Documents/SpaMTP_test_data/", mass.range = c(160,1500), resolution = 10,
```

loadSpaMTPDatabase	<i>Load versioned SpaMTP annotation resources</i>
--------------------	---

Description

Loads a coherent group of annotation resources from [SpaMTPdb](#). Retrieved resources are cached for the current R session. A named custom bundle can be supplied for offline, testing, or user-curated workflows.

Usage

```
loadSpaMTPDatabase(
  resources = c("chem_props", "source_df", "analyte", "analytehaspathway", "pathway"),
  version = "latest",
  source = c("auto", "spamtpdb"),
  database = NULL,
  local_dir = NULL,
  hub = NULL,
  offline = FALSE,
  refresh = FALSE,
  verify = TRUE
)
```

Arguments

resources	Character vector of resource names. Use <code>spaMTPDatabaseInfo()</code> to list valid names.
version	SpaMTPdb/RaMP resource version, or "latest".
source	Database source. "auto" and "spamtpdb" resolve versioned resources through SpaMTPdb.
database	Optional named list containing the requested resources. When supplied, no Hub lookup is performed.
local_dir	Optional directory containing staged SpaMTPdb .rds files.
hub	Optional pre-created AnnotationHub object passed to SpaMTPdb.
offline	If TRUE, do not query AnnotationHub.
refresh	If TRUE, bypass SpaMTP's in-session resource cache.
verify	Verify local resource files against the SpaMTPdb registry. Defaults to TRUE. Set to FALSE only for intentional development fixtures; use database for a named custom bundle instead of presenting modified resources as an official snapshot.

Details

Exact affected RaMP 3.0.7 topology resources receive a checksum-guarded correction for historical interaction-code and direction recycling. Source labels are retained in `source_reaction_type`; the resource attribute `spamtp_interaction_repair` records the correction separately from the original download metadata. Published files are unchanged. See `vignette("Pathway_Database_Integration", package = "SpaMTP")` for provenance and reproducible build instructions.

Value

A named list containing the requested resources.

Examples

```
utils::str(formals(loadSpaMTPDatabase))
example_database <- list(
  ramp_db_metadata = list(ramp_version = "example")
)
database <- loadSpaMTPDatabase(
  "ramp_db_metadata",
  database = example_database
)
names(database)
```

malDiMatrixProfiles *List MALDI matrix and on-tissue derivatization profiles*

Description

The registry separates conventional MALDI matrices, reactive matrices, and derivatization reagents. A registered profile does not necessarily imply that a universal mass-shift rule is available: some reactions require a coupling reagent, produce several products, or remain strongly dependent on the analyte structure and acquisition conditions.

Usage

```
malDiMatrixProfiles(matrix = NULL)
```

Arguments

matrix Optional matrix/reagent name or alias. When NULL, return the complete registry.

Value

A data frame describing supported matrix profiles and their current automatic-rule status.

Examples

```
utils::str(formals(malDiMatrixProfiles))
```

malDiMatrixRules *Select annotation rules from the MALDI matrix/reagent profile*

Description

Conventional matrices select ordinary ion rules and, where validated, matrix-metabolite adducts. Reactive matrices additionally select covalent product rules. Profiles without a verified universal net transformation are retained in the registry but do not generate speculative mass shifts.

Usage

```
malDiMatrixRules(
  malDi_matrix,
  polarity = NULL,
  include_standard = TRUE,
  include_matrix_products = TRUE
)
```

Arguments

- maldi_matrix** Matrix or derivatization reagent name; aliases such as "2,5-DHB", "HCCA", "9-AA", and "FMP-10" are accepted.
- polarity** "positive", "negative", or "neutral". If NULL, use the profile default; profiles supporting both modes require an explicit choice.
- include_standard**
Include the validated ordinary adduct rules.
- include_matrix_products**
Include validated matrix-adduct or reactive product rules.

Value

A validated adduct/reaction rule data frame accepted by `buildMZAnnotationIndex()`.

Examples

```
utils::str(formals(maldiMatrixRules))
```

mapGeneIdentifiers	<i>Map gene identifiers with an auditable HGNC identity</i>
--------------------	---

Description

Accepts approved, previous and alias symbols (bare or `gene_symbol:`), HGNC, Entrez, Ensembl gene IDs (optional version suffix), UniProt and RaMP IDs. Ambiguous identifiers are excluded, never expanded to multiple genes. Different RaMP records of the same resolved HGNC gene share one canonical RaMP ID. Gene references are supplied by SpaMTPdb; this function does not query live annotation services. Input order and duplicates are preserved.

Usage

```
mapGeneIdentifiers(
  genes,
  database = NULL,
  index = NULL,
  gene_reference = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  ambiguous = c("exclude", "error"),
  verbose = TRUE,
  database_version = "latest",
  database_local_dir = NULL,
  offline = FALSE
)
```

Arguments

genes	Vector of gene identifiers.
database	Optional named SpaMTPdb resource bundle with source_df.
index	Optional reusable index from buildGeneMappingIndex() . If provided, its recorded source and reference determine the mapping. If database is also supplied, its source-table fingerprint must match.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spaMTPdbGeneRef . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
ambiguous	Exclude ambiguous mappings with audit status, or error.
verbose	Warn about unresolved input rows.
database_version	RaMP version, independent of the HGNC version.
database_local_dir	Local staged RaMP directory.
offline	Require local or cached HGNC data.

Value

One row per input with gene_id (HGNC), approved symbol, canonical_ramp_id, all equivalent ramp_ids (semicolon separated), status, mapping method and candidate_gene_ids. The gene_mapping attribute records provenance. mapped_ramp_only explicitly denotes an unharmonized RaMP record without a resolved HGNC identity. candidate_ramp_ids records raw candidates encountered for direct RaMP IDs or source crossreferences, including unresolved/conflicting records. A shared crossreference with unresolved competitors remains ambiguous even if only one candidate has a resolved HGNC identity.

Examples

```
utils::str(formals(mapGeneIdentifiers))
```

mapSpatialOmics

*Map metabolite intensities to transcriptomic pixels or cells***Description**

Both inputs must already use the same planar coordinate units and matching sample_id values. Mapping never links different samples. ST.hires uses target centroids inside square MSI pixels; ties use the nearest MSI centre then pixel ID. Otherwise each MSI pixel must cover overlap.threshold of the target spot/polygon area. Matching MSI intensities are averaged with equal weights.

Usage

```
mapSpatialOmics(
  SM.data,
  ST.data,
  ST.hires = FALSE,
  SM.assay = "main",
  ST.assay = "main",
  SM.pixel.width = NULL,
  overlap.threshold = 0.2,
  annotations = TRUE,
  add.metadata = TRUE,
  merge.unique.metadata = TRUE,
  map.data = FALSE,
  new_SPT.assay = "transcriptome",
  new_SPM.assay = "main",
  verbose = FALSE,
  ST.radius = NULL,
  ST.polygons = NULL,
  dropUnmapped = FALSE
)
```

Arguments

SM.data, ST.data	SpatialExperiment objects with aligned coordinates.
ST.hires	Use point-in-pixel mapping for high-resolution target cells.
SM.assay, ST.assay	Primary ("main") or alternative experiment names in the source and target containers; see experimentAccess . These select modalities, not individual expression matrices.
SM.pixel.width	Square MSI pixel width in coordinate units. NULL estimates the median positive nearest-neighbour distance within samples.
overlap.threshold	Minimum fraction of target area covered by a pixel.
annotations	Preserve MSI rowData and annotation provenance.

add.metadata	Include summaries of source pixel metadata in colData.
merge.unique.metadata	Remove duplicate values in source summaries.
map.data	Map all source assays; FALSE maps counts only.
new_SPT.assay	Name of the transcriptome alternative experiment.
new_SPM.assay	Name of the primary metabolomics experiment.
verbose	Show progress messages.
ST.radius	Target spot radius, scalar or one per target pixel, in spatial coordinate units; alternatively supply ST.polygons.
ST.polygons	An sf polygon object with a cell column matching ST.data.
dropUnmapped	Remove targets without matching MSI pixels. FALSE keeps zero-filled intensities and marks these pixels with mapped = FALSE.

Value

A SpatialExperiment with MSI assays on the target coordinates, target transcriptomes in altExp, target images and metadata, and mapping provenance.

massIntensityPlot	<i>Plot mean mass spectra from a native spatial experiment</i>
-------------------	--

Description

Feature masses come from rowData, so arbitrary feature IDs are supported.

Usage

```
massIntensityPlot(
  data,
  group.by = NULL,
  split.by = NULL,
  cols = NULL,
  assay = "main",
  slot = "counts",
  label.annotations = FALSE,
  annotation.column = "all_IsomerNames",
  mz.labels = NULL,
  metabolite.labels = NULL,
  xlab = "m/z",
  ylab = "intensity",
  mass.range = NULL,
  y.lim = NULL,
  labelCex = 5,
  labelAdj = -1,
  labelOffset = 0,
```

```

    labelCol = "#eb4034",
    nlabels.to.show = NULL
  )

```

Arguments

<code>data</code>	A SpatialExperiment or aligned Cardinal experiment.
<code>group.by</code>	Optional colData column defining overlaid groups.
<code>split.by</code>	Optional colData column defining separate panels; mutually exclusive with <code>group.by</code> .
<code>cols</code>	Optional group colours.
<code>assay</code>	Primary or alternative experiment.
<code>slot</code>	Expression assay. See experimentAccess for the experiment/matrix distinction.
<code>label.annotations</code>	Display annotations instead of m/z labels.
<code>annotation.column</code>	Annotation column in rowData.
<code>mz.labels</code>	Numeric m/z values to label using nearest measured peaks.
<code>metabolite.labels</code>	Metabolite names to label; mutually exclusive with <code>mz.labels</code> .
<code>xlab, ylab</code>	Axis labels.
<code>mass.range</code>	Optional two-value displayed mass range.
<code>y.lim</code>	Optional y-axis limits.
<code>labelCex</code>	Text size.
<code>labelAdj, labelOffset</code>	Vertical and horizontal text justification.
<code>labelCol</code>	Text colour.
<code>nlabels.to.show</code>	Maximum annotations displayed per feature.

Value

A ggplot.

Examples

```
utils::str(formals(massIntensityPlot))
```

metaspaceClient	<i>Metaspace R Client Constructor</i>
-----------------	---------------------------------------

Description

Creates a client object with methods to interact with the METASPACE GraphQL API.

Usage

```
metaspaceClient(host = "https://metaspace2020.org", api_key = NULL)
```

Arguments

host	Character string specifying the METASPACE host URL (default = "https://metaspace2020.org").
api_key	Optional. Your METASPACE API key for accessing private datasets. If accessing a public dataset this can be left as NULL (default = NULL).

Value

A list containing functions for querying METASPACE data.

Examples

```
utils::str(formals(metaspaceClient))
# ms <- metaspaceClient()
```

metaspaceToFeatureMatrix	<i>Convert METASPACE Images to Feature Matrix</i>
--------------------------	---

Description

Transforms the list of 2D ion image matrices from getMetaspace into a single matrix suitable for spatial analysis packages.

Usage

```
metaspaceToFeatureMatrix(metaspace_data, transform = FALSE, verbose = TRUE)
```

Arguments

metaspace_data	A list object returned by getMetaspace.
transform	Logical. If TRUE, transposes each image matrix before flattening (e.g., to align coordinate systems) (default = FALSE).
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Value

A numeric matrix where rows are molecular features (m/z) and columns are spatial pixels (named 'x_y').

Examples

```
utils::str(formals(metaspacerToFeatureMatrix))
# metaspacerToFeatureMatrix(mtx, transform = TRUE)
```

multiOmicIntegration *Integrate paired modalities using native PCA embeddings*

Description

SpatialExperiment input uses paired alternative experiments and combines equal-weight PCA embeddings generated with scater. This is not Seurat WNN; the historical WNN workflow remains in the published-workflow branch.

Usage

```
multiOmicIntegration(
  multiomic.data,
  reduction.list = list("spm.pca", "spt.pca"),
  dims.list = list(1:30, 1:30),
  return.intermediate = FALSE,
  verbose = FALSE,
  modalities = c("main", "transcriptome"),
  ...
)

## S4 method for signature 'SpatialExperiment'
multiOmicIntegration(
  multiomic.data,
  reduction.list = list("spm.pca", "spt.pca"),
  dims.list = list(1:30, 1:30),
  return.intermediate = FALSE,
  verbose = FALSE,
  modalities = c("main", "transcriptome"),
  ...
)

## S4 method for signature 'ANY'
multiOmicIntegration(
  multiomic.data,
  reduction.list = list("spm.pca", "spt.pca"),
  dims.list = list(1:30, 1:30),
  return.intermediate = FALSE,
```

```

    verbose = FALSE,
    modalities = c("main", "transcriptome"),
    ...
  )

```

Arguments

multiomic.data A SpatialExperiment with a primary modality and paired alternative experiments in `SingleCellExperiment::altExps()`.

reduction.list Reduction names for the primary MSI and alternative modalities (default = `list("spm.pca", "spt.pca")`).

dims.list List containing the numeric range of principle component dimension to include for each modality (default = `list(1:30, 1:30)`).

return.intermediate Retain per-modality PCA results in `reducedDims()`.

verbose Show progress messages (default = `FALSE`).

modalities Primary experiment ("main") followed by alternative experiments to integrate. Derived pathway and merged experiments are not included automatically.

... Additional arguments passed to `scater::runPCA()`.

Value

The input container with an integrated representation. For SpatialExperiment this is `reducedDim(x, "integrated")`.

Examples

```

utils::str(formals(multiOmicIntegration))
# SpaMTP.obj <- multiOmicIntegration(SpaMTP.obj, reduction.list = list("spt.pca", "spm.pca"), dims.list = list(1:

```

mzBoxPlot

Generates a Boxplot of spatial metabolic intensity data

Description

Generates a Boxplot of spatial metabolic intensity data

Usage

```

mzBoxPlot(
  data,
  group.by = NULL,
  mzs = NULL,
  assay = "main",
  slot = "counts",
  title = "BoxPlot",

```

```

x.lab = "var",
y.lab = "intensity",
show.points = TRUE,
bottom.cutoff = NULL,
top.cutoff = NULL,
log.data = FALSE,
cols = NULL,
verbose = FALSE
)

```

Arguments

<code>data</code>	Bioconductor experiment containing the metabolomic intensity data.
<code>group.by</code>	Name of the <code>colData()</code> column to group by (default = <code>NULL</code>).
<code>mzs</code>	Vector of characters defining which features (m/z's) to label on the plot. If <code>NULL</code> no features will be labeled (default = <code>NULL</code>).
<code>assay</code>	Primary (main) or alternative experiment name.
<code>slot</code>	Expression assay name within the selected experiment. See experimentAccess for the experiment/matrix distinction.
<code>title</code>	Character string of the plot title (default = "BoxPlot").
<code>x.lab</code>	Character string of the x-axis label (default = "var").
<code>y.lab</code>	Character string of the y-axis label (default = "intensity").
<code>show.points</code>	Boolean value describing whether to show each individual data point (default = <code>TRUE</code>).
<code>bottom.cutoff</code>	Numeric value defining the percent of data to exclude for the lower end of the distribution. A <code>bottom.cutoff = 0.05</code> will remove the bottom 5% of data point (default = <code>NULL</code>).
<code>top.cutoff</code>	Numeric value defining the percent of data to exclude for the upper end of the distribution. A <code>top.cutoff = 0.05</code> will remove the top 5% of data point (default = <code>NULL</code>).
<code>log.data</code>	Boolean value indicating whether to log transform the y-axis values (default = <code>FALSE</code>).
<code>cols</code>	Vector of strings defining the colours to use for plotting. This vector should match the length of unique groups (default = <code>NULL</code>).
<code>verbose</code>	Boolean indicating whether to show the message. If <code>TRUE</code> the message will be show, else the message will be suppressed (default = <code>FALSE</code>).

Value

A ggplot box-plot object.

Examples

```
utils::str(formals(mzBoxPlot))
```

mzRidgePlot*Generates a ridge plot of spatial metabolic intensity data*

Description

Generates a ridge plot of spatial metabolic intensity data

Usage

```
mzRidgePlot(  
  data,  
  group.by = NULL,  
  mzs = NULL,  
  assay = "main",  
  slot = "counts",  
  title = "RidgePlot",  
  x.lab = "intensity",  
  y.lab = "var",  
  bottom.cutoff = NULL,  
  top.cutoff = NULL,  
  bins = 1000,  
  log.data = FALSE,  
  cols = NULL,  
  verbose = FALSE  
)
```

Arguments

data	Bioconductor experiment containing the metabolomic intensity data.
group.by	Name of the colData() column to group by (default = NULL).
mzs	Vector of characters defining which features (m/z's) to label on the plot. If NULL no features will be labeled (default = NULL).
assay	Primary (main) or alternative experiment name.
slot	Expression assay name within the selected experiment. See experimentAccess for the experiment/matrix distinction.
title	Character string of the plot title (default = "RidgePlot").
x.lab	Character string of the x-axis label (default = "var").
y.lab	Character string of the y-axis label (default = "intensity").
bottom.cutoff	Numeric value defining the percent of data to exclude for the lower end of the distribution. A bottom.cutoff = 0.05 will remove the bottom 5% of data point (default = NULL).
top.cutoff	Numeric value defining the percent of data to exclude for the upper end of the distribution. A top.cutoff = 0.05 will remove the top 5% of data point (default = NULL).

bins	number of bins to group
log.data	Boolean value indicating whether to log transform the y-axis values (default = FALSE).
cols	Vector of strings defining the colours to use for plotting. This vector should match the length of unique groups (default = NULL).
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = FALSE).

Value

A ggplot ridge-plot object.

Examples

```
utils::str(formals(mzRidgePlot))
```

mzViolinPlot

Generates a violin plot of spatial metabolic intensity data

Description

Generates a violin plot of spatial metabolic intensity data

Usage

```
mzViolinPlot(
  data,
  group.by = NULL,
  mzs = NULL,
  assay = "main",
  slot = "counts",
  title = "Expression",
  x.lab = "var",
  y.lab = "intensity",
  show.points = TRUE,
  bottom.cutoff = NULL,
  top.cutoff = NULL,
  log.data = FALSE,
  cols = NULL,
  verbose = FALSE
)
```

Arguments

<code>data</code>	Bioconductor experiment containing the metabolomic intensity data.
<code>group.by</code>	Name of the <code>colData()</code> column to group by (default = <code>NULL</code>).
<code>mzs</code>	Vector of characters defining which features (m/z's) to label on the plot. If <code>NULL</code> no features will be labeled (default = <code>NULL</code>).
<code>assay</code>	Primary (main) or alternative experiment name.
<code>slot</code>	Expression assay name within the selected experiment. See experimentAccess for the experiment/matrix distinction.
<code>title</code>	Character string of the plot title (default = "Expression").
<code>x.lab</code>	Character string of the x-axis label (default = "var").
<code>y.lab</code>	Character string of the y-axis label (default = "intensity").
<code>show.points</code>	Boolean value describing whether to show each individual data point (default = <code>TRUE</code>).
<code>bottom.cutoff</code>	Numeric value defining the percent of data to exclude for the lower end of the distribution. A <code>bottom.cutoff = 0.05</code> will remove the bottom 5% of data point (default = <code>NULL</code>).
<code>top.cutoff</code>	Numeric value defining the percent of data to exclude for the upper end of the distribution. A <code>top.cutoff = 0.05</code> will remove the top 5% of data point (default = <code>NULL</code>).
<code>log.data</code>	Boolean value indicating whether to log transform the y-axis values (default = <code>FALSE</code>).
<code>cols</code>	Vector of strings defining the colours to use for plotting. This vector should match the length of unique groups (default = <code>NULL</code>).
<code>verbose</code>	Boolean indicating whether to show the message. If <code>TRUE</code> the message will be show, else the message will be suppressed (default = <code>FALSE</code>).

Value

A ggplot violin-plot object.

Examples

```
utils::str(formals(mzViolinPlot))
```

normalizeSMData

Normalize spatial metabolomics intensity data

Description

SingleCellExperiment methods store the result in the standard `normcounts` or `logcounts` assays. Cardinal input delegates TIC normalization to Cardinal and remains file-backed. Convert Seurat inputs explicitly before analysis. Cardinal normalization follows Cardinal's deferred-processing model; queued steps are executed by `Cardinal::process()`, `binSpaMTP()`, or conversion of an aligned experiment to `SpatialExperiment`.

Usage

```
normalizeSMDData(  
  data,  
  normalisation.type = "TIC",  
  scale.factor = NULL,  
  assay = "main",  
  slot = "counts",  
  verbose = TRUE  
)  
  
## S4 method for signature 'SingleCellExperiment'  
normalizeSMDData(  
  data,  
  normalisation.type = "TIC",  
  scale.factor = NULL,  
  assay = "main",  
  slot = "counts",  
  verbose = TRUE  
)  
  
## S4 method for signature 'MSImagingExperiment'  
normalizeSMDData(  
  data,  
  normalisation.type = "TIC",  
  scale.factor = NULL,  
  assay = "main",  
  slot = "counts",  
  verbose = TRUE  
)  
  
## S4 method for signature 'MSImagingArrays'  
normalizeSMDData(  
  data,  
  normalisation.type = "TIC",  
  scale.factor = NULL,  
  assay = "main",  
  slot = "counts",  
  verbose = TRUE  
)  
  
## S4 method for signature 'ANY'  
normalizeSMDData(  
  data,  
  normalisation.type = "TIC",  
  scale.factor = NULL,  
  assay = "main",  
  slot = "counts",  
  verbose = TRUE  
)
```

)

Arguments

data	A SingleCellExperiment (including SpatialExperiment), Cardinal MSImagingArrays/MSImagingExperiment.
normalisation.type	"TIC" (default) or "RC" divides each pixel's intensities by its total and multiplies by scale.factor, storing normcounts. "LogNormalize" additionally applies natural log1p() and stores logcounts. Cardinal input supports only "TIC".
scale.factor	Positive target total before log transformation. When NULL, native TIC uses the number of features and RC/LogNormalize use 10000. Set to 1e6 explicitly for counts/intensities per million. Zero-total pixels remain zero. Cardinal requires NULL and uses its own TIC scaling.
assay	Primary (main) or alternative experiment name.
slot	Expression assay name within the selected experiment. See experimentAccess for the experiment/matrix distinction.
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Value

An object of the same container family with normalized values.

Examples

```
utils::str(formals(normalizeSMDData))
```

pathwayNetworkPlots *Construct an interactive pathway network visualization*

Description

Builds a self-contained data payload and an interactive D3 visualization for selected pathways across spatial clusters. Topology lookup, edge preparation, differential-expression matching, and matrix extraction are indexed so they are not repeated for every cluster-pathway pair.

Usage

```
pathwayNetworkPlots(
  SpaMTP,
  ident,
  regpathway,
  DE.list,
  selected_pathways = NULL,
  path = getwd(),
```

```

SM_slot = "counts",
ST_slot = "counts",
colour_palette = NULL,
SM_assay = "main",
ST_assay = "transcriptome",
analyte_types = c("genes", "metabolites"),
annotation_source = c("current", "auto", "legacy"),
annotation_score_threshold = NULL,
annotation_score_floor = 0.01,
metabolite_detection = c("leading_edge", "annotated"),
image = NULL,
verbose = TRUE,
top_n_pathways = 10L,
max_nodes = 500L,
label_mode = c("detected", "all", "none"),
max_spatial_points = 50000L,
layout_mode = c("repulsion", "force", "radial", "bipartite"),
database = NULL,
database_version = "latest",
database_source = c("auto", "spamtpdb"),
database_local_dir = NULL,
gene_mapping = c("auto", "hgnc", "ramp"),
gene_reference = NULL,
gene_index = NULL,
gene_reference_version = "latest",
gene_reference_local_dir = NULL,
organism = "Homo sapiens",
pathway_index = NULL
)

```

Arguments

SpaMTP	A SpatialExperiment with paired transcriptomics in altExp. Subset to one sample before exporting. Metabolomics must be annotated by annotateSM() , or supplied as explicit compound IDs from createPathwayAssay() .
ident	Metadata column used to identify spatial clusters or regions.
regpathway	Data frame returned by findRegionalPathways() .
DE.list	One differential-expression data frame per requested analyte type. Data frames must contain cluster, gene, avg_log2FC (or logFC), and p_val_adj (or FDR). Descriptive native marker tables containing mean_auc and mean_cohen are also supported; their P values remain unavailable. A named list is recommended.
selected_pathways	Optional pathway names or source IDs. Matching is case-insensitive. When NULL, the most important pathways are selected by summed absolute NES.
path	Output directory for the generated HTML file.
SM_slot	Expression assay name within the SM_assay experiment. See experimentAccess for the experiment/matrix distinction.

ST_slot	Expression assay name within the ST_assay experiment. See experimentAccess for the experiment/matrix distinction.
colour_palette	Colours used for the spatial abundance raster.
SM_assay	Primary ("main") or alternative experiment containing MSI.
ST_assay	Paired transcriptome alternative experiment name.
analyte_types	One or both of "genes" and "metabolites".
annotation_source	Metabolite annotation provenance. The default, "current", requires the indexed, scored RaMP annotation output. "auto" permits a warned legacy fallback and "legacy" requests it explicitly.
annotation_score_threshold	Minimum indexed annotation score used for metabolite matching. When NULL, the threshold recorded by <code>findRegionalPathways()</code> is reused (default = NULL).
annotation_score_floor	Lowest annotation score embedded in the HTML for interactive filtering. It is automatically lowered when the initial <code>annotation_score_threshold</code> is smaller (default = 0.01).
metabolite_detection	Which metabolite nodes receive detected styling. "leading_edge" uses only GSEA leading-edge metabolites; "annotated" also includes every score-filtered annotation belonging to the selected pathway, regardless of DE significance.
image	Optional image ID validated against <code>imgData</code> . Coordinates always come from <code>spatialCoords</code> ; the optical raster is not embedded in this viewer.
verbose	Display progress messages.
top_n_pathways	Number of pathways selected when <code>selected_pathways</code> is NULL.
max_nodes	Maximum number of nodes per cluster-pathway view. Detected leading-edge analytes are always retained, followed by their neighbours and high-degree nodes. Use Inf to disable pruning.
label_mode	Initial node-label mode: detected nodes only, all nodes, or no labels. It can also be changed interactively.
max_spatial_points	Maximum number of spatial points embedded in the HTML. Larger datasets are deterministically downsampled for responsive browser rendering. Use Inf to retain all points.
layout_mode	Initial network layout. "repulsion" maximises spacing, "force" uses a balanced force-directed layout, "radial" separates analyte classes into rings, and "bipartite" separates genes and metabolites vertically. It can also be changed interactively.
database	Optional named list of database resources, normally created by loadSpaMTPDatabase() .
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see loadSpaMTPDatabase() .

database_local_dir	Optional staged SpaMTPdb resource directory.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through <code>SpaMTPdb : spaMTPdbGeneReference</code> . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by <code>buildGeneMappingIndex()</code> . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
pathway_index	A reusable index from <code>buildPathwayIndex()</code> . When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.

Details

Gene differential values use the shared HGNC mapping index. For display, the same resolved gene value is attached to its equivalent raw RaMP topology nodes; this does not introduce additional enrichment tests.

Value

Invisibly returns the generated HTML file path.

Examples

```
utils::str(formals(pathwayNetworkPlots))
# pathwayNetworkPlots(
#   SpaMTP, ident = "Custom_ident", regpathway = regpathway,
#   DE.list = DE.list, selected_pathways = "WP1902"
# )
```

pixelPlot	<i>Convert point symbols to squares without changing coordinates or scales</i>
-----------	--

Description

For combined feature panels, request combine=FALSE from the original plot function and pass the resulting list. This function changes point symbols, not tissue geometry or the physical pixel footprint.

Usage

```
pixelPlot(plot)
```

Arguments

plot	A ggplot with point layers, or a list of such plots.
------	--

Value

The ggplot or list, retaining images, coordinates, scales and labels.

Examples

```
utils::str(formals(pixelPlot))
```

plot3DFeature	<i>Compare two spatial features in 3D</i>
---------------	---

Description

Expression comes from the primary or alternative experiment; numeric colData columns can also be plotted. Coordinates and images use SpatialExperiment accessors. Only one sample is displayed at a time.

Usage

```
plot3DFeature(
  data,
  features,
  assays = c("transcriptome", "main"),
  slots = "counts",
  between.layer.height = 100,
  names = NULL,
  size = 3,
  col.palette = "Reds",
```

```

x.axis.label = "x",
y.axis.label = "y",
z.axis.label = "z",
show.x.ticks = FALSE,
show.y.ticks = FALSE,
show.z.ticks = FALSE,
show.image = NULL,
plot.height = 800,
plot.width = 1500,
image.sf = NULL,
downscale.image = NULL,
sampleId = NULL
)

```

Arguments

<code>data</code>	A SpatialExperiment or aligned Cardinal experiment.
<code>features</code>	One or two feature IDs or numeric colData column names. A single feature is repeated for comparison across two experiments.
<code>assays</code>	One or two primary/altExp names.
<code>slots</code>	One or two expression assay names. See experimentAccess for the experiment/matrix distinction.
<code>between.layer.height</code>	Distance between layers.
<code>names</code>	Optional layer labels.
<code>size</code>	Marker size.
<code>col.palette</code>	One or two Plotly colour palettes.
<code>x.axis.label, y.axis.label, z.axis.label</code>	Axis labels.
<code>show.x.ticks, show.y.ticks, show.z.ticks</code>	Show axis tick labels.
<code>show.image</code>	Optional image ID from imgData.
<code>plot.height, plot.width</code>	Widget dimensions.
<code>image.sf</code>	Retired Seurat scale-factor selector; must be NULL. The selected imgData row supplies the scaleFactor.
<code>downscale.image</code>	Optional positive integer raster sampling stride.
<code>sampleId</code>	Sample ID; required when data contains multiple samples.

Value

A Plotly widget.

Examples

```
utils::str(formals(plot3DFeature))
```

plotPathways

*Plot named pathways on a dimensionality reduction***Description**

Plot named pathways on a dimensionality reduction

Usage

```
plotPathways(
  pathways,
  object,
  title = NULL,
  assay = "main",
  slot = "logcounts",
  reduction = NULL,
  colors = c("darkblue", "lightgrey", "darkred"),
  guide = "colourbar",
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  pathway_index = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  duplicate_genes = c("error", "mean", "sum"),
  ...
)
```

Arguments

pathways	Character vector of pathwayRampId values or unambiguous database pathway names.
object	A SpatialExperiment with reducedDims.
title	Optional plot title.
assay	Primary ("main") or alternative experiment name.
slot	Expression assay name within the selected experiment (default = "logcounts"), read with SummarizedExperiment::assay(). See experimentAccess for the experiment/matrix distinction.
reduction	Name in reducedDimNames(object). NULL prefers UMAP, TSNE, then PCA, otherwise the first available reduction.

colors	Gradient colours.
guide	Colour guide type.
database	Optional named list of database tables.
database_version	Database version.
database_source	Database source, auto or spamtpdb.
database_local_dir	Optional local database directory.
pathway_index	A reusable index from buildPathwayIndex(). When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spamtpdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.
...	Additional arguments to <code>scater::plotReducedDim</code> .

Value

A named list of ggplots with pathway_coverage and pathway_index attributes; each plot also carries its exact pathway_scores.

plotPathwaysSpatially *Plot named pathways in tissue coordinates*

Description

Plot named pathways in tissue coordinates

Usage

```
plotPathwaysSpatially(
  pathways,
  object,
  images = NULL,
  title = NULL,
  image.alpha = 1,
  assay = "main",
  slot = "logcounts",
  colors = c("darkblue", "lightgrey", "darkred"),
  guide = "colourbar",
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  pathway_index = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  duplicate_genes = c("error", "mean", "sum"),
  ...
)
```

Arguments

pathways	Character vector of pathwayRampId values or unambiguous database pathway names.
object	A SpatialExperiment with reducedDims.
images	Image IDs in imgData to overlay; NULL draws coordinates only.
title	Optional plot title.
image.alpha	Image opacity.
assay	Primary ("main") or alternative experiment name.
slot	Expression assay name within the selected experiment (default = "logcounts"), read with SummarizedExperiment::assay(). See experimentAccess for the experiment/matrix distinction.

colors	Gradient colours.
guide	Colour guide type.
database	Optional named list of database tables.
database_version	Database version.
database_source	Database source, auto or spamtpdb.
database_local_dir	Optional local database directory.
pathway_index	A reusable index from buildPathwayIndex(). When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spamtpdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.
...	Additional arguments to plotSinglePathwaySpatially.

Value

A named list of spatial pathway plots, carrying the same coverage and score attributes as plotPathways().

plotRegionalPathways *Plot significantly enriched pathways per region*

Description

Visualisation of Set Enrichment Analysis Results from `SpaMTP::findRegionalPathways()`.

Usage

```
plotRegionalPathways(
  regpathway,
  ident.column = "Cluster_id",
  selected_pathways = NULL,
  sig_cutoff = NULL,
  num_display = NULL,
  text_size = NULL,
  verbose = TRUE,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL
)
```

Arguments

<code>regpathway</code>	A dataframe generated by the <code>SpaMTP::findRegionalPathways()</code> function, containing identified regional pathways.
<code>ident.column</code>	A character string specifying the column name of the <code>regpathway</code> dataframe containing the ids or clusters to compare (default = "Cluster_id").
<code>selected_pathways</code>	A character vector specifying the names or IDs of pathways to be included in the analysis (e.g., <code>c("Amino acid metabolism", "WP1902", "Aspartate and asparagine metabolism")</code>). This argument is not case-sensitive (default = NULL).
<code>sig_cutoff</code>	Within-family BH-adjusted P-value cutoff for the point outline. NULL uses 0.05. Display selection does not recompute adjustment.
<code>num_display</code>	An integer specifying the number of pathways to display in the plot. If set to null will plot the smaller of either 10 pathways or the number of unique pathways provided in 'regpathway' (default = NULL).
<code>text_size</code>	A numeric value controlling the size of the text elements in the plot. If NULL the default text size is 12 (default = NULL).
<code>verbose</code>	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
<code>database</code>	Optional named list of database resources, normally created by loadSpaMTPDatabase() .

database_version
SpaMTPdb/RaMP version used for pathway lookup.

database_source
Database source; see [loadSpaMTPDatabase\(\)](#).

database_local_dir
Optional staged SpaMTPdb resource directory.

Value

A ggplot object representing the set enrichment analysis results.

Examples

```
utils::str(formals(plotRegionalPathways))
# plotRegionalPathways(SpaMTP, ident = "clusters", regpathway = pathway_df)
```

plotSinglePathway	<i>Plot a pathway score on a Bioconductor dimensionality reduction</i>
-------------------	--

Description

Scores are the sum of matched feature values divided by the square root of the matched feature count, then centred and scaled across pixels. Constant scores are shown as zero. Plotting uses `scater::plotReducedDim`.

Usage

```
plotSinglePathway(
  pathway,
  object,
  title = NULL,
  assay = "main",
  slot = "logcounts",
  reduction = NULL,
  colors = c("darkblue", "lightgrey", "darkred"),
  guide = "colourbar",
  pathway_index = NULL,
  duplicate_genes = "error",
  ...
)
```

Arguments

pathway	Character vector of feature IDs in one pathway.
object	A SpatialExperiment with reducedDims.
title	Optional plot title.
assay	Primary ("main") or alternative experiment name.

slot	Expression assay name within the selected experiment (default = "logcounts"), read with <code>SummarizedExperiment::assay()</code> . See experimentAccess for the experiment/matrix distinction.
reduction	Name in <code>reducedDimNames(object)</code> . NULL prefers UMAP, TSNE, then PCA, otherwise the first available reduction.
colors	Gradient colours.
guide	Colour guide type.
pathway_index	A reusable index from <code>buildPathwayIndex()</code> . When supplied, its gene identities and memberships must match any explicit database or <code>gene_index</code> . Fingerprints prevent incompatible reuse.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.
...	Additional arguments to <code>scater::plotReducedDim</code> .

Value

A ggplot displaying the pathway score.

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(logcounts = rbind(a = 1:4, b = 4:1)),
  spatialCoords = cbind(x = 1:4, y = 0),
  colData = S4Vectors::DataFrame(row.names = paste0("p", 1:4)))
SingleCellExperiment::reducedDim(x, "PCA") <- cbind(1:4, c(1, 3, 2, 4))
plotSinglePathway("a", x)
```

plotSinglePathwaySpatially

Plot a pathway score in tissue coordinates

Description

Plot a pathway score in tissue coordinates

Usage

```
plotSinglePathwaySpatially(
  pathway,
  object,
  images = NULL,
  title = NULL,
  image.alpha = 1,
```

```

assay = "main",
slot = "logcounts",
colors = c("darkblue", "lightgrey", "darkred"),
guide = "colourbar",
crop = TRUE,
min.cutoff = NA,
max.cutoff = NA,
ncol = NULL,
pt.size.factor = 1.6,
alpha = 1,
shape = 16,
stroke = 0,
interactive = FALSE,
image.labels = NULL,
pathway_index = NULL,
duplicate_genes = "error"
)

```

Arguments

<code>pathway</code>	Character vector of feature IDs in one pathway.
<code>object</code>	A <code>SpatialExperiment</code> with <code>reducedDims</code> .
<code>images</code>	Image IDs in <code>imgData</code> to overlay; <code>NULL</code> draws coordinates only.
<code>title</code>	Optional plot title.
<code>image.alpha</code>	Image opacity.
<code>assay</code>	Primary ("main") or alternative experiment name.
<code>slot</code>	Expression assay name within the selected experiment (default = "logcounts"), read with <code>SummarizedExperiment::assay()</code> . See experimentAccess for the experiment/matrix distinction.
<code>colors</code>	Gradient colours.
<code>guide</code>	Colour guide type.
<code>crop</code>	Whether to restrict plotting to the pixel extent.
<code>min.cutoff, max.cutoff</code>	Numeric or quantile colour cutoffs.
<code>ncol</code>	Number of columns in the combined plot.
<code>pt.size.factor</code>	Point size.
<code>alpha</code>	Point opacity (the first value is used).
<code>shape</code>	Point shape.
<code>stroke</code>	Point border width.
<code>interactive</code>	Convert the result to an interactive plotly plot.
<code>image.labels</code>	Optional labels for the selected images.
<code>pathway_index</code>	A reusable index from <code>buildPathwayIndex()</code> . When supplied, its gene identities and memberships must match any explicit database or <code>gene_index</code> . Fingerprints prevent incompatible reuse.

duplicate_genes

How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.

Value

A ggplot or combined plot; a plotly object if interactive is TRUE.

Examples

```
x <- SpatialExperiment::SpatialExperiment(
  assays = list(logcounts = rbind(a = 1:4, b = 4:1)),
  spatialCoords = cbind(x = 1:4, y = 0),
  colData = S4Vectors::DataFrame(row.names = paste0("p", 1:4)))
plotSinglePathwaySpatially("a", x)
```

plotSpatialFeature	<i>Plot spatial features from a Bioconductor container</i>
--------------------	--

Description

This S4 plotting entry point reads feature values from an assay and pixel positions from `SpatialExperiment::spatialCoords`. It therefore does not require a Seurat image or field-of-view object.

Usage

```
plotSpatialFeature(
  x,
  features,
  assayName = "counts",
  labels = NULL,
  pointSize = 1.6,
  alpha = 1,
  minCutoff = NA,
  maxCutoff = NA,
  combine = TRUE,
  ncol = NULL,
  ...
)

## S4 method for signature 'SpatialExperiment'
plotSpatialFeature(
  x,
  features,
  assayName = "counts",
  labels = NULL,
```

```

    pointSize = 1.6,
    alpha = 1,
    minCutoff = NA,
    maxCutoff = NA,
    combine = TRUE,
    ncol = NULL,
    ...
)

## S4 method for signature 'MSImagingExperiment'
plotSpatialFeature(
  x,
  features,
  assayName = "counts",
  labels = NULL,
  pointSize = 1.6,
  alpha = 1,
  minCutoff = NA,
  maxCutoff = NA,
  combine = TRUE,
  ncol = NULL,
  ...
)

```

Arguments

<code>x</code>	A spatial experiment.
<code>features</code>	Feature names to plot.
<code>assayName</code>	Assay containing the feature values.
<code>labels</code>	Optional panel labels corresponding to features.
<code>pointSize</code>	Point size.
<code>alpha</code>	Point opacity.
<code>minCutoff, maxCutoff</code>	Numeric cutoffs, or strings such as "q05" and "q95", recycled over features.
<code>combine</code>	Whether to combine multiple panels.
<code>ncol</code>	Number of columns in a combined plot.
<code>...</code>	Reserved for class-specific methods.

Value

A ggplot, a combined plot, or a list of plots.

Examples

```
methods::showMethods("plotSpatialFeature")
```

ppm_error	<i>Calculates the ppm error as a value</i>
-----------	--

Description

Calculates the ppm error as a value

Usage

```
ppm_error(observed_mz, reference_mz, ppm)
```

Arguments

observed_mz	Numeric value defining the observed mz value.
reference_mz	Numeric value defining the reference mz value.
ppm	Numeric value defining the maximum acceptable ppm_error/threshold for searching.

Value

Numeric value defining the ppm_error between the observed and reference mz value

Examples

```
### Helper Function ###
```

ppm_range_match	<i>Check whether m/z values match within a ppm tolerance</i>
-----------------	--

Description

Calculates the absolute ppm difference between observed and reference m/z values and returns whether each comparison satisfies the tolerance.

Usage

```
ppm_range_match(observed_mz, reference_mz, ppm)
```

Arguments

observed_mz	Numeric value defining the observed mz value.
reference_mz	Numeric value defining the reference mz value.
ppm	Numeric value defining the maximum acceptable ppm_error/threshold for searching.

Value

Boolean value indicating if a match is found (TRUE) or not (FALSE)

Examples

```
### Helper Function ###
```

```
predictAdductsFromSMILES
```

Predict a structure-aware adduct search space from SMILES

Description

This function applies the same functional-group and ion-mode logic used by [buildMZAnnotationIndex\(\)](#) before any observed m/z is supplied. It is useful for auditing which protonation, deprotonation, alkali-binding, or reactive matrix hypotheses SpaMTP will retain for a structure.

Usage

```
predictAdductsFromSMILES(  
  smiles,  
  polarity = NULL,  
  maldi_matrix = NULL,  
  rules = NULL,  
  min_structure_score = 0.05,  
  backend = c("auto", "native"),  
  workers = getOption("SpaMTP.smiles_workers", 1L)  
)
```

Arguments

smiles	Character vector of SMILES strings.
polarity	"positive", "negative", or "neutral". When NULL, use the matrix-profile default or positive mode.
malDI_matrix	Optional MALDI matrix/reagent profile.
rules	Optional custom rule table. It cannot be combined with maldi_matrix.
min_structure_score	Minimum rule-specific structural prior marked as retained.
backend, workers	Passed to deconvolveSMILES() .

Value

A ranked data frame with one row per structure/rule combination.

Examples

```
utils::str(formals(predictAdductsFromSMILES))
```

print.spamtp_mz_index *Print an indexed metabolite annotation search space*

Description

Print an indexed metabolite annotation search space

Usage

```
## S3 method for class 'spamtp_mz_index'
print(x, ...)
```

Arguments

x A spamtp_mz_index returned by [buildMZAnnotationIndex\(\)](#).
... Additional arguments reserved for print-method compatibility.

Value

x, invisibly.

Pseudo_msms	<i>Find all library spectra10 above a cosine threshold for each imaging peak, restricting to a global precursor-mz range.</i>
-------------	---

Description

Find all library spectra10 above a cosine threshold for each imaging peak, restricting to a global precursor-mz range.

Usage

```
Pseudo_msms(  
  raw_mz,  
  counts,  
  spectra10_list,  
  ppm_tol = 10,  
  frag_tol_da = 0.01,  
  cos_threshold = 0.7,  
  min_precursor = 0,  
  max_precursor = Inf  
)
```

Arguments

raw_mz	Numeric vector of length n_peaks
counts	Numeric matrix n_peaks × n_pixels
spectra10_list	List of Spectrum2 objects (length N_lib)
ppm_tol	Precursor-matching tolerance in ppm (default 10)
frag_tol_da	Fragment-matching tolerance in Da (default 0.01)
cos_threshold	Minimum cosine similarity to report (default 0.7)
min_precursor	Minimum allowed precursor m/z (default = 0, i.e. no lower bound)
max_precursor	Maximum allowed precursor m/z (default = Inf, i.e. no upper bound)

Value

A list of length n_peaks. Each element is an integer vector giving the indices in spectra10_list of all library spectra10 with cosine >= cos_threshold (or integer(0) if none).

queryMZAnnotationIndex

Query an indexed metabolite annotation search space

Description

Candidates are pruned by ppm tolerance and proton/charge validity during index construction, then ranked using mass error, rule prior, mass-defect, isotope, and contextual adduct-family evidence.

Usage

```
queryMZAnnotationIndex(
  observed_mz,
  index,
  ppm = 5,
  ms1_spectrum = NULL,
  use_mass_defect = TRUE,
  check_isotopes = TRUE,
  check_adduct_network = TRUE,
  min_score = 0
)
```

Arguments

observed_mz	Numeric vector of observed m/z values.
index	A spamtp_mz_index from buildMZAnnotationIndex() .
ppm	Mass tolerance in parts per million.
ms1_spectrum	Optional contextual spectrum with mz and intensity columns. It should represent the same retention-time window or spatial pixel/region as the queried peaks.

use_mass_defect Apply the CHO negative mass-defect penalty.
check_isotopes Score carbon-13 and, when relevant, chlorine/bromine M+2 patterns.
check_adduct_network Downweight complex ions whose base monomer family is absent from `ms1_spectrum`.
min_score Minimum final score to retain.

Value

A ranked candidate data frame.

Examples

```
utils::str(formals(queryMZAnnotationIndex))
```

reaction_type	<i>Pathway-network reaction styles</i>
---------------	--

Description

A small lookup table mapping SpaMTP interaction style codes to visual properties used by the interactive pathway-network viewer.

Usage

```
data(reaction_type)
```

Format

A data frame with 18 rows and 5 variables:

reaction_type Stable SpaMTP style code; not a database-specific factor index.

reaction_name Human-readable reaction type.

linetype Line style.

arrowhead Arrowhead style.

colour Edge colour.

Value

A data frame of pathway-network reaction styles.

readSMMatrix

*Read a binned spatial metabolomics matrix***Description**

This function reads SM data stored in a table format, with one row per pixel and two columns named 'x' and 'y' storing its coordinates. All other columns should be the relative m/z values containing the intensity values of each pixel. Coordinates are stored only in `spatialCoords()`, not duplicated in pixel metadata where a later alignment could leave them out of date.

Usage

```
readSMMatrix(
  mtx.file,
  assay = "Spatial",
  verbose = TRUE,
  feature.start.column = 1,
  mz.prefix = NULL,
  project.name = "SpaMTP",
  returnType = "SpatialExperiment"
)
```

Arguments

<code>mtx.file</code>	Character string defining the path of the spatial metabolomic image matrix .csv file.
<code>assay</code>	Label stored in <code>mainExpName()</code> ; intensities are in the counts assay.
<code>verbose</code>	Boolean indicating whether to show the message. If TRUE, the message will be shown; else, it will be suppressed (default = TRUE).
<code>feature.start.column</code>	Numeric value defining the start index containing the x, y, and m/z value columns within the table (default = 1).
<code>mz.prefix</code>	Character string matching the prefix string in front of each m/z name (default = NULL).
<code>project.name</code>	Character string defining the name of the sample to be assigned as <code>orig.ident</code> (default = "SpaMTP").
<code>returnType</code>	Output container; only "SpatialExperiment" is supported.

Details

NOTE: The input file must be in a format similar to the table below:

```
A data.frame: 5 × 5
  x  y  mz1  mz2  mz3
1  0  1    0    0  11
```

2	0	2	0	0	0
3	0	3	0	0	0
4	0	4	20	0	0
5	0	5	0	0	0

- The first two columns (x, y) contain the respective spatial coordinates.
- The subsequent columns contain the m/z values and their intensities for each spatial pixel.

Value

A SpatialExperiment.

Examples

```
utils::str(formals(readSMMatrix))
# msi_data <- readSMMatrix("~/Documents/msi_mtx.csv")
```

refineLipids	<i>Uses common lipid nomenclature to simplify lipid annotations</i>
--------------	---

Description

Based on rgoslin, this function has the ability to simplify lipid annotations into more general terms, groups and classes. This function can input a data frame containing a column storing the annotation and refine only the entries which are annotated as known lipid structures.

Usage

```
refineLipids(
  data,
  annotation.column = "annotations",
  database = "HMDB",
  lipid_info = "simple"
)
```

Arguments

data	data.frame containing a column with lipid annotations
annotation.column	Character string matching the column name containing the lipid annotations in the provided df (default = "annotations").
database	Character string defining the database matching the lipid annotations. Possible entries include c('Shorthand2020','Goslin','FattyAcids','LipidMaps','SwissLipids','HMDB') (default = "HMDB").
lipid_info	Character string defining what level of information to return for each lipid. Options are either "all" or "simple". "Simple" returns a smaller, simplified list of annotations for each lipid. Please visit https://bioconductor.org/packages/release/bioc/vignettes/rgoslin/inst/ to see all possible outputs (default = "simple").

Value

Data.frame containing additional columns with simplified lipid names

Examples

```
utils::str(formals(refineLipids))
# refineLipids(DEPs_df)
```

renderSpaMTPReport	<i>Render a portable, self-contained multi-omics report</i>
--------------------	---

Description

The HTML embeds PNG figures and styles and needs no network, Pandoc or R session to view. Full tables, mapping weights, processing settings and the analysed object are also saved for reproducibility. Existing files are never overwritten. A report is published to output_dir only after rendering succeeds.

Usage

```
renderSpaMTPReport(
  result,
  output_dir,
  max_points = 5000,
  interactive = TRUE,
  preview_features = 180,
  preview_points = min(max_points, 2000),
  table_rows = 1000
)
```

Arguments

result	A spamtp_workflow result from runSpaMTPWorkflow.
output_dir	New or empty output directory.
max_points	Maximum observations displayed per scatter plot. Analyses and exported tables are not subsampled.
interactive	Embed an offline region/DE explorer using local JavaScript.
preview_features	Maximum feature profiles embedded in the explorer.
preview_points	Maximum spatial/distribution observations embedded.
table_rows	Maximum preloaded ranked rows per region or contrast. Full tables are exported independently of this preview limit.

Value

Absolute path to report.html, invisibly.

Examples

```
utils::str(formals(renderSpaMTPReport))
```

runDE	<i>Run limma contrasts on log-transformed technical pools</i>
-------	---

Description

Worker function for calculating differentially abundant metabolites per pooling group. Used by the compatibility method = "technical" branch of [findAllDEMs\(\)](#). Fits limma to $\log_2(\text{counts}(\text{pooled_data}) + 1)$ and uses `limma::treat()`; an edgeR DGEList stores the values and results, but no edgeR count-model test or voom transformation is performed. Technical pools are not independent biological replicates.

Usage

```
runDE(
  pooled_data,
  data,
  ident,
  output_dir,
  run_name,
  n,
  logFC_threshold,
  annotation.column,
  assay,
  return.individual = FALSE,
  verbose = TRUE
)
```

Arguments

pooled_data	A SingleCellExperiment object which contains the pooled pseudo-replicate data.
data	The original SummarizedExperiment, including SingleCellExperiment or SpatialExperiment, supplying pixel groups and feature annotations.
ident	Column in <code>colData(data)</code> and <code>colData(pooled_data)</code> defining groups.
output_dir	Existing directory for output CSV tables, or NULL to omit file output.
run_name	A character string defining the title of this DE analysis (will be used when saving DEMs to .csv file).
n	An integer that defines the number of pseudo-replicates per sample (default = 3).
logFC_threshold	Fold-change threshold passed as <code>log2(logFC_threshold)</code> to <code>limma::treat()</code> (default = 1.2).

annotation.column	Column in the selected experiment's rowData() containing feature annotations, or NULL to omit annotations.
assay	Primary ("main") or alternative experiment name in data. Pooled expression is always read from counts(pooled_data).
return.individual	Boolean value defining whether to return a list of individual edgeR objects for each designated ident. If FALSE, one merged edgeR object will be returned (default = FALSE).
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Value

A modified edgeR object which contains the relative pseudo-bulking analysis outputs, including a DEMs data.frame with a list of differential expressed m/z metabolites

Examples

```
utils::str(formals(runDE))
```

runMetabolicPCA	<i>Compute PCA with scater on a selected experiment</i>
-----------------	---

Description

Uses scater::runPCA() and stores embeddings in reducedDim(). The user can provide a bin/resolution size to increase the bin size and reduce the dimensionality/noise of the SM dataset prior to calculating PCAs.

Usage

```
runMetabolicPCA(
  SpaMTP,
  npcs = 30,
  variance_explained_threshold = 0.9,
  assay = "main",
  slot = "counts",
  show_variance_plot = FALSE,
  bin_resolution = NULL,
  resolution_units = "ppm",
  bin_method = "sum",
  reduction.name = "pca",
  verbose = TRUE,
  features = NULL,
  scale = FALSE
)
```

Arguments

SpaMTP	A SingleCellExperiment, including SpatialExperiment, containing the selected primary or alternative experiment.
npcs	Positive number of PCs, capped by matrix dimensions. NULL uses the variance threshold.
variance_explained_threshold	Cumulative variance fraction in (0, 1], used only when npcs is NULL.
assay	Primary (main) or alternative experiment name.
slot	Expression assay name within the selected experiment (default = "counts"), read with SummarizedExperiment::assay(). See experimentAccess for the experiment/matrix distinction.
show_variance_plot	Boolean indicating weather to display the variance plot output by this analysis (default = FALSE).
bin_resolution	Numeric value defining the resolution to use for binning m/z peaks. If set to NULL, no binning will be performed (default = NULL).
resolution_units	Character string specifying the units of the bin_resolution. Either 'ppm' or 'mz' can be provided. bin_resolution must be provided for this parameter to be implemented (default = "ppm").
bin_method	Character string defining the method to use for binning respective m/z peaks that fall within a bin. Options for this parameter can be one of "sum", "mean", "max" or "min". bin_resolution must be provided for this parameter to be implemented (default = "sum").
reduction.name	Name in SingleCellExperiment::reducedDimNames(SpaMTP) for the resulting embedding (default = "pca"). Even when the input expression is in an alternative experiment, the embedding is stored in the parent object's reducedDim().
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).
features	Optional exact feature identifiers. When supplied, all and only these rows enter PCA; the scater top-500 default is bypassed.
scale	Scale features to unit variance before PCA.

Value

The input with PCA embeddings in reducedDim().

Examples

```
counts <- matrix(seq_len(30), nrow = 5)
sce <- SingleCellExperiment::SingleCellExperiment(assays = list(counts = counts))
sce <- runMetabolicPCA(sce, npcs = 2)
SingleCellExperiment::reducedDimNames(sce)
## For running PCA on un-adjusted peak bin sizes
# spamtp_obj <- runMetabolicPCA(spamtp_obj, npcs = 50)
```

```
## For running PCA on increased peak bin sizes
# spamtp_obj <- runMetabolicPCA(spamtp_obj, npcs = 50, bin_resolution = 50)
```

runPooling	<i>Pools Bioconductor experiment into random pools for pseudo-bulking.</i>
------------	--

Description

Runs pooling of a SpaMTP dataset to generate pseudo-replicates for each unique identity provided. This function is used by `findAllDEMs()`. These random pools are technical partitions, not independent biological replicates; resulting p-values must not be interpreted as biological replication. For population inference aggregate independent samples instead.

Usage

```
runPooling(data.filt, idents, n, assay, slot, seed = 1234, verbose = TRUE)
```

Arguments

<code>data.filt</code>	A Bioconductor experiment containing count values for pooling.
<code>idents</code>	A character string defining the idents column to pool the data against.
<code>n</code>	An integer defining the amount of pseudo-replicates to generate for each sample (default = 3).
<code>assay</code>	Primary ("main") or alternative experiment name.
<code>slot</code>	Expression assay name within the selected experiment. Values must be finite and non-negative for technical pooling. See experimentAccess for the experiment/matrix distinction.
<code>seed</code>	Numeric value used to set the seed for reproducible randomisation (default = 1234).
<code>verbose</code>	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Value

A `SingleCellExperiment` object which contains pooled (n)-pseudo-replicate counts data based on the Bioconductor experiment input

Examples

```
utils::str(formals(runPooling))
```

runRAMPGeseca	<i>Runs multilevel Monte-Carlo variant for performing gene sets co-regulation analysis using the RAMP_DB metabolite/gene database.</i>
---------------	--

Description

This function is adapted from the `fgsea::geseca` package to identify significantly expressed RAMP_DB pathways based on an expression/feature embedding matrix.

Usage

```
runRAMPGeseca(
  E,
  minSize = 1,
  maxSize = nrow(E) - 1,
  center = TRUE,
  scale = FALSE,
  sampleSize = 101,
  eps = 1e-50,
  nproc = 0,
  BPPARAM = NULL,
  nPermSimple = 1000,
  database = NULL,
  database_version = "latest",
  database_source = c("auto", "spamtpdb"),
  database_local_dir = NULL,
  pathway_index = NULL,
  gene_mapping = c("auto", "hgnc", "ramp"),
  gene_reference = NULL,
  gene_index = NULL,
  gene_reference_version = "latest",
  gene_reference_local_dir = NULL,
  organism = "Homo sapiens",
  duplicate_genes = c("error", "mean", "sum")
)
```

Arguments

E	Expression matrix with gene identifiers or RaMP IDs as rows and observations as columns.
minSize	Minimal size of a gene set to test. All pathways below the threshold are excluded (default = 1).
maxSize	Maximal size of a gene set to test. All pathways above the threshold are excluded (default = nrow(E) - 1).
center	a logical value indicating whether the gene expression should be centered to have zero mean before the analysis takes place (default = TRUE).

scale	a logical value indicating whether the gene expression should be scaled to have unit variance before the analysis takes place (default = FALSE).
sampleSize	sample size for conditional sampling (default = 101).
eps	This parameter sets the boundary for calculating P-values (default = 1e-50).
nproc	If not equal to zero sets BPPARAM to use nproc workers (default = 0).
BPPARAM	Parallelization parameter used in bplapply (default = NULL).
nPermSimple	Number of permutations in the simple geseca implementation for preliminary estimation of P-values (default = 1000).
database	Optional named list of database resources, normally created by loadSpaMTPDatabase() .
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
pathway_index	A reusable index from buildPathwayIndex(). When supplied, its gene identities and memberships must match any explicit database or gene_index. Fingerprints prevent incompatible reuse.
gene_mapping	"auto" uses HGNC identity harmonization for official SpaMTPdb resources and for custom bundles with gene_reference or gene_index. "hgnc" requires harmonization; "ramp" explicitly retains RaMP-only mapping for historical reproduction or a curated non-human database. Custom bundles without a reference use RaMP-only mapping in auto mode and never trigger a reference download.
gene_reference	HGNC reference data frame. NULL retrieves the pinned reference through SpaMTPdb : spaMTPdbGeneReference . Custom references require unique hgnc_id and symbol; optional fields are status, entrez_id, ensembl_gene_id, uniprot_ids, prev_symbol, alias_symbol.
gene_index	Optional index built from the same source table by buildGeneMappingIndex() . Multiple RaMP records of a resolved HGNC gene have their pathway memberships united and count once. Conflicting RaMP records and ambiguous symbols are excluded and recorded in the audit.
gene_reference_version	Version in SpaMTPdb's separate HGNC registry.
gene_reference_local_dir	Local HGNC resource directory.
organism	Must be "Homo sapiens"; this index is not a mouse-to-human orthology map. Use a separately curated species-specific database for other organisms.
duplicate_genes	How to aggregate expression rows of one gene. The default errors for differing rows and deduplicates identical rows. Explicit mean or sum requires choosing an appropriate expression scale; sum is generally inappropriate for log-transformed expression.

Value

A table with GESECA results. Each row corresponds to a tested RAMP_DB pathway.

Examples

```
utils::str(formals(runRAMPGeseca))
# sig_pathways <- runRAMPGeseca(E, minSize=15, maxSize=500)
```

runSpaMTPWorkflow	<i>Run a reproducible spatial multi-omics workflow and HTML report</i>
-------------------	--

Description

Acquires native inputs, audits QC, aligns and maps independent modalities, normalizes explicitly selected layers, computes per-modality PCA and an equal-weight joint representation, and produces descriptive associations, optional replicate-level contrasts, annotations and shared-index pathways. See the `End_to_End_Workflow` vignette for the input and configuration schema.

Usage

```
runSpaMTPWorkflow(
  input,
  modalities,
  output_dir = NULL,
  reference = NULL,
  alignment = list(),
  mapping = list(),
  group = NULL,
  replicate = NULL,
  contrasts = list(),
  observation_unit = "spatial observation",
  coordinate_units = NULL,
  npcs = 15,
  max_features = 2000,
  clusters = NULL,
  association_features = 20,
  seed = 1,
  title = "SpaMTP multi-omics report",
  regions = group,
  native = list(),
  structure = list()
)
```

Arguments

<code>input</code>	A <code>SpatialExperiment</code> , its RDS path, a pinned <code>SpaMTPData</code> resource specification, or a named list of independent inputs of these forms.
<code>modalities</code>	Named list of specifications with type and layer, plus optional normalization, <code>scale_factor</code> , <code>min_detected</code> , <code>species</code> , <code>annotation</code> , and <code>pathways</code> . Paired input names are main followed by selected <code>altExp</code> names.

output_dir	Optional new directory for report, tables and result RDS.
reference	Name of the reference modality for independent inputs.
alignment	Named settings for each moving modality: method='identity' explicitly declares registered coordinates; otherwise arguments for applySpatialAlignment (e.g. method='affine', alignment or landmarks).
mapping	Named per-moving-modality specifications: method='nearest' with max_distance, or method='pixel' with width and optional target_radius, target_polygons and overlap_threshold, in aligned coordinate units.
group	Optional colData field for descriptive summaries and plot colour.
replicate	Optional colData field identifying biological replicates.
contrasts	Named list of numerator, denominator and paired settings. Tests require at least three independent biological replicates per group.
observation_unit	Description of what a column represents.
coordinate_units	Description of reference coordinate units; required when independent inputs are mapped.
npcs, max_features	Maximum PCs and variable features per modality.
clusters	Optional explicit k for exploratory k-means in joint space, or in PCA space when exactly one modality is selected. NULL does not choose a biological partition automatically.
association_features	Maximum regional-marker/variable-feature anchors for each modality pair. Targets are the other modality's PCA-selected features. Raw and covariate-residual correlations have no pixel P values.
seed	Random seed, restored on exit.
title	Report title.
regions	Region metadata column, defaulting to group. NULL can use requested workflow clusters when group is absent.
native	Named native spatial settings; see analyzeSpaMTPRegions.
structure	Native representation/evaluation settings: spatial, primary (pca or spatial), neighbors, lambda, platform, external reference metadata, k_grid, stability_blocks and umap. Graph PCA uses all retained observations and the same selected, scaled features as ordinary PCA. Clustering stability conditions on full-data embeddings; it is not independent-specimen validation.

Details

Only observations passing QC in every selected modality and having mappings in every moving modality enter joint analysis. Excluded positions remain in the audit. Raw input assays are preserved. Automatic normalization uses $\log_2(1 + 10000 * \text{value} / \text{column sum})$ for metabolomics/transcriptomics counts, $\log_2(1 + \text{value})$ for protein counts, and preserves other layers. This is a configurable baseline, not a batch correction or platform-specific raw-signal preprocessing method. Spatial mapping can reuse source pixels; correlations and clusters are exploratory. Contrasts average workflow values within biological replicate and group, not random pixel pools.

Value

A spatmp_workflow list containing object, QC, alignment/mapping audit, analysis, settings, stage log and session information. output_dir also receives a self-contained report.html, CSV tables and workflow.rds.

Examples

```
utils::str(formals(runSpaMTPWorkflow))
```

runSpatialGraphPCA	<i>Perform Dimensionality Reduction using Graph-Regularised PCA on Spatial Data</i>
--------------------	---

Description

Computes a graph-regularised PCA using spatial coordinates and scaled expression data. A k-nearest neighbour (k-NN) graph is computed using spatial locations and used to regularise the PCA decomposition via a graph Laplacian. The result is stored in reducedDim(), with spatial edges in colPairs().

Usage

```
runSpatialGraphPCA(
  data,
  n_components = 50,
  assay = "main",
  slot = "scaled",
  image = NULL,
  platform = "Visium",
  lambda = 0.5,
  n_neighbors = NULL,
  include_self = FALSE,
  alg = "kd_tree",
  fast = TRUE,
  graph_name = "SpatialKNN",
  reduction_name = "SpatialPCA",
  verbose = TRUE
)
```

Arguments

data	A SpatialExperiment or aligned Cardinal MSIImagingExperiment.
n_components	Integer specifying the number of principal components to compute (default = 50).
assay	Primary (main) or alternative experiment name.

slot	Expression assay name within the selected experiment (default = "scaled"), read with <code>SummarizedExperiment::assay()</code> . See experimentAccess for the experiment/matrix distinction.
image	Reserved; must be NULL. Subset pixels before running graph PCA.
platform	Character string matching either "Visium" or "ST" to determine how the k-NN graph is constructed. If "Visium" k-nns will handle the hexagon spot arrangement, including setting <code>n_neighbors = 6</code> , else "ST" assignment will set <code>n_neighbors = 4</code> unless a value is specifically provided (default = "Visium").
lambda	Numeric value defining the regularisation parameter that controls the influence of the graph Laplacian (default = 0.5).
n_neighbors	Integer value specifying the number of spatial neighbours to use. If NULL, will default of 6 for "Visium" data and 4 for "ST" platforms (default = NULL).
include_self	Boolean logical value indicating whether to include self-connections in the graph (default = FALSE).
alg	Character string specifying the algorithm to use for nearest neighbour search (passed to <code>FNN::get.knn()</code>) (default = "kd_tree").
fast	Boolean logical value stating whether to use fast approximate eigendecomposition via <code>RSpectra::eigs_sym()</code> . For large datasets this is recommended (default = TRUE).
graph_name	Name for pixel edges in <code>SingleCellExperiment::colPair()</code> and the graph matrix in <code>S4Vectors::metadata(data)\$spatialGraphs</code> (default = "SpatialKNN").
reduction_name	Name for the parent object's <code>SingleCellExperiment::reducedDim()</code> (default = "SpatialPCA").
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Details

Note: This method has been adapted from the GraphPCA Python package ([doi:10.1186/s13059-02403429x](https://doi.org/10.1186/s13059-02403429x)).

Value

A `SpatialExperiment` with a new `colPair()` graph and `reducedDim()` embedding. Expression assays are preserved.

Examples

```
utils::str(formals(runSpatialGraphPCA))
# spamtp_obj <- runSpatialGraphPCA(spamtp_obj, platform = "Visium")
```

savePheatmapAsPDF	<i>Saves a demsHeatmap as a PDF</i>
-------------------	-------------------------------------

Description

Saves a demsHeatmap as a PDF

Usage

```
savePheatmapAsPDF(pheatmap, filename, width = 20, height = 20)
```

Arguments

pheatmap	A pheatmap plot object that is being saved.
filename	Character string defining the full filepath and name of the plot to be saved as.
width	Integer value representing the width of the saved pdf plot (default = 20).
height	Integer value representing the height of the saved pdf plot (default = 20).

Value

The generated PDF path, invisibly.

Examples

```
utils::str(formals(savePheatmapAsPDF))
# savePheatmapAsPDF(pheatmap, filename = "/Documents/plots/pheatmap1")
```

saveSpaMTPData	<i>Export an experiment's expression matrix and metadata</i>
----------------	--

Description

This function saves a Bioconductor experiment into a standard single-cell/spatial file format. This includes a filtered_feature_bc_matrix folder containing files storing the features, barcode/pixels and intensity matrix. Pixel metadata are also stored. Feature annotations and spatial image files are optional exports; the export does not preserve every alternative experiment or analysis result. Use saveRDS() for an entire R container.

Usage

```
saveSpaMTPData(
  data,
  outdir,
  assay = "Spatial",
  slot = "counts",
  image = NULL,
  annotations = FALSE,
  generate.h5 = TRUE,
  verbose = TRUE
)
```

Arguments

data	A SummarizedExperiment, including SingleCellExperiment or SpatialExperiment. Image export requires SpatialExperiment.
outdir	Character string of the directory to save the mtx.mtx, barcode.tsv, features.tsv, barcode_metadata.csv and feature_metadata.csv in.
assay	Character string defining the primary or alternative experiment that contains the m/z count data (default = "Spatial").
slot	Expression assay name within the selected experiment (default = "counts"), read with SummarizedExperiment::assay(). See experimentAccess for the experiment/matrix distinction.
image	Image ID identifying exactly one imgData row; NULL omits image export.
annotations	Boolean values defining if the Bioconductor experiment contains annotations to be saved (default = FALSE).
generate.h5	Boolean value indicating whether to generate a filtered_feature_bc_matrix.h5 file. Often used by data loading functions (e.g. scanpy.load_visium). If FALSE, only a filtered_feature_bc_matrix folder will be generated (default = TRUE).
verbose	Boolean indicating whether to show informative processing messages. If TRUE the message will be show, else the message will be suppressed (default = TRUE).

Details:

- This can be used for saving data for transfer to python. Can be read in as Anndata using scanpy.read_10x_mtx().
- For saving in R saveRDS() is recommended.

Value

The output-directory path, invisibly.

Examples

```
utils::str(formals(saveSpaMTPData))
```

scaleSMData

*Centre and scale features within a Bioconductor experiment***Description**

Each feature is centred and/or scaled across pixels using base R's `scale()`. Zero-variance features become zero after centring and scaling. The input assay is preserved and the result is stored in scaled by default. This materialises a dense matrix; use implicit centring in PCA instead when a full scaled matrix would exceed available memory.

Usage

```
scaleSMData(
  data,
  assay = "main",
  slot = "logcounts",
  center = TRUE,
  scale = TRUE,
  outputAssay = "scaled"
)
```

Arguments

<code>data</code>	A <code>SingleCellExperiment</code> , including <code>SpatialExperiment</code> .
<code>assay</code>	Primary ("main") or alternative experiment name.
<code>slot</code>	Input expression assay, usually "logcounts". See experimentAccess for the experiment/matrix distinction.
<code>center, scale</code>	Logical values passed to base <code>scale()</code> .
<code>outputAssay</code>	Name of a new assay for scaled values. Must differ from the input assay and must not already exist.

Value

The input container with a scaled assay in the selected experiment.

Examples

```
counts <- matrix(seq_len(20), nrow = 4,
  dimnames = list(paste0("mz-", 101:104), paste0("p", 1:5)))
sce <- SingleCellExperiment::SingleCellExperiment(assays = list(counts = counts))
sce <- normalizeSMDData(sce, "LogNormalize", verbose = FALSE)
sce <- scaleSMData(sce)
SummarizedExperiment::assayNames(sce)
```

searchAnnotations	<i>Find Annotation</i>
-------------------	------------------------

Description

Searches through annotated m/z values to return all which contain the metabolite search term provided

Usage

```
searchAnnotations(  
  data,  
  metabolite,  
  assay = "main",  
  search.exact = FALSE,  
  column.name = "all_IsomerNames"  
)
```

Arguments

data	Bioconductor experiment containing annotated m/z values.
metabolite	Character string of metabolite search term.
assay	Character string defining the primary or alternative experiment that contains the annotated metadata corresponding to the m/z values (default = "Spatial").
search.exact	Boolean value defining if to only return m/z values which contain the exact match to the metabolite search term (default = FALSE).
column.name	Character string defining the feature-metadata column where annotations are stored (default = "all_IsomerNames").

Value

A Data.Frame containing the peak metadata corresponding to the metabolite search term provided

Examples

```
utils::str(formals(searchAnnotations))
```

selectROIs*Launch an Interactive ROI Annotation App for SpatialExperiment*

Description

This function opens a Shiny app to allow users to manually annotate Regions of Interest (ROIs) on spatial transcriptomics or metabolomics data stored in a `SpatialExperiment`. Users can interactively select regions using lasso selection, assign custom names to each ROI, and save the results as binary (0/1) columns in `colData`.

Usage

```
selectROIs(object, sampleId = NULL, launch = TRUE)
```

Arguments

<code>object</code>	A <code>SpatialExperiment</code> with spatial coordinates and metadata.
<code>sampleId</code>	The sample to select regions from; required for multiple samples.
<code>launch</code>	Launch the app; <code>FALSE</code> returns a Shiny app for embedding or testing.

Details

The app includes options to:

- Choose a metadata column to display.
- Adjust the spot size for display.
- Use lasso to select spots and name each ROI.
- Save each ROI to `colData` without overwriting existing columns.
- Return the final `SpatialExperiment` upon clicking "Finish".

Selection uses the exact points returned by `plotly`'s lasso tool.

Value

A `SpatialExperiment` with ROI columns (1 = selected, 0 = unselected within sample, NA = other samples), or a Shiny app if `launch` is `FALSE`.

Examples

```
utils::str(formals(selectROIs))
```

seuratToSingleCellExperiment

Convert Seurat expression data to SingleCellExperiment

Description

Conversion is the only part of SpaMTP that uses SeuratObject. Coordinates are not required. The selected layer keeps its original name (for example, counts or data); select that assay explicitly in downstream functions. Other Seurat assays with the same pixels become altExp() entries. Only one exact layer is converted, not all split layers. Join split layers in Seurat before conversion when the complete dataset is needed. Cell/feature metadata, active identities and stored analysis metadata are retained. Reductions, graphs, optical images and Seurat commands are not converted. Stored metadata should contain portable R/Bioconductor values; entries containing Seurat objects/classes in lists are skipped with a warning.

Usage

```
seuratToSingleCellExperiment(
  x,
  assay = NULL,
  layer = "counts",
  includeAltExps = TRUE
)
```

Arguments

x	A Seurat object.
assay	Primary Seurat assay; defaults to its active assay.
layer	Exact layer to convert. No partial matching is performed.
includeAltExps	Convert other assays with the same pixels.

Value

A SingleCellExperiment with the selected expression layer.

Examples

```
if (requireNamespace("SeuratObject", quietly = TRUE)) {
  counts <- matrix(seq_len(12), nrow = 3,
    dimnames = list(paste0("gene", 1:3), paste0("cell", 1:4)))
  seu <- SeuratObject::CreateSeuratObject(counts)
  sce <- seuratToSingleCellExperiment(seu)
  SummarizedExperiment::assayNames(sce)
}
```

`seuratToSpatialExperiment`*Convert a Seurat object to SpatialExperiment*

Description

Convert once at the workflow boundary, then use Bioconductor containers throughout SpaMTP. Expression and metadata follow `seuratToSingleCellExperiment()`. Coordinates are matched by pixel names, never by position. Supply coordinates for multi-image data or image formats without x/y columns; coordinate units and orientation must already be consistent. Optical rasters, graphs and reductions are not transferred. Attach optical images separately with `addSpatialImage()`. When a single image/FOV is present, its public `GetTissueCoordinates()` accessor is the default coordinate source, even if historical x/y columns remain in cell metadata. Metadata coordinates are used automatically only when there are no images. Multiple images always require an explicit selection. Centroids are read as pixel centres, not expanded polygon vertices. The selected source is recorded in `metadata(x)$seurat_interoperability`; retained legacy coordinate columns are archival and must not replace `spatialCoords(x)` in native analysis.

Usage

```
seuratToSpatialExperiment(  
  x,  
  assay = NULL,  
  layer = "counts",  
  image = NULL,  
  includeAltExps = TRUE,  
  coordinates = NULL  
)
```

Arguments

<code>x</code>	A Seurat object.
<code>assay</code>	Primary Seurat assay; defaults to its active assay.
<code>layer</code>	Exact layer to convert, usually "counts".
<code>image</code>	Optional Seurat image/FOV used for coordinates. A single image is selected automatically; its coordinates take precedence over metadata.
<code>includeAltExps</code>	Convert other assays with matching pixels.
<code>coordinates</code>	Optional matrix/data.frame with x/y columns and pixel names in row names or a cell column. Cannot be combined with image.

Value

A `SpatialExperiment`.

Examples

```

if (requireNamespace("SeuratObject", quietly = TRUE)) {
  counts <- matrix(seq_len(12), nrow = 3,
    dimnames = list(paste0("mz-", 101:103), paste0("pixel", 1:4)))
  seu <- SeuratObject::CreateSeuratObject(counts)
  xy <- cbind(x = 1:4, y = c(0, 1, 0, 1))
  rownames(xy) <- colnames(seu)
  spe <- seuratToSpatialExperiment(seu, coordinates = xy)
  spe <- normalizeSMDData(spe, verbose = FALSE)
}

```

spaMTPDatabaseInfo	<i>Inspect SpaMTP database resources</i>
--------------------	--

Description

Inspect SpaMTP database resources

Usage

```
spaMTPDatabaseInfo(version = NULL)
```

Arguments

version	Optional SpaMTPdb resource version. NULL lists every available external version.
---------	--

Value

A data frame describing resources in SpaMTPdb.

Examples

```

utils::str(formals(spaMTPDatabaseInfo))
spaMTPDatabaseInfo()

```

spatialExperimentToCardinal	<i>Convert a SpatialExperiment to Cardinal</i>
-----------------------------	--

Description

Convert a SpatialExperiment to Cardinal

Usage

```
spatialExperimentToCardinal(x, assayName = "counts")
```

Arguments

x	A SpatialExperiment containing aligned MSI features.
assayName	Name of the assay to convert.

Value

A Cardinal MSImagingExperiment.

Examples

```
utils::str(formals(spatialExperimentToCardinal))
```

```
spatialExperimentToSeurat
```

Convert a SpatialExperiment to an optional Seurat object

Description

This conversion is provided only for users who explicitly need a Seurat workflow. Only SeuratObject is needed for conversion; Seurat analysis is not used. The requested layer is placed in the Seurat counts layer; choose counts when raw intensities are needed. Other layers, reductions and optical images are not converted; spatial coordinates are exported as centroids. Alternative experiments without the requested layer are skipped with a warning, rather than treating a derived score assay as raw counts.

Usage

```
spatialExperimentToSeurat(
  x,
  assayName = "Spatial",
  layer = "counts",
  imageName = "fov",
  includeAltExps = TRUE
)
```

Arguments

x	A SpatialExperiment.
assayName	Name for the primary Seurat assay.
layer	SpatialExperiment assay to convert.
imageName	Name for the centroid FOV.
includeAltExps	Convert alternative experiments to Seurat assays.

Value

A Seurat object.

Examples

```
utils::str(formals(spatialExperimentToSeurat))
```

```
spatialMZAnnotationPlot
```

Plot annotated metabolites with optional optical images

Description

Matching peaks are summed using the same rules as imageMZAnnotationPlot.

Usage

```
spatialMZAnnotationPlot(
  object,
  metabolites,
  plusminus = NULL,
  images = NULL,
  crop = TRUE,
  assay = "main",
  slot = "counts",
  keep.scale = "feature",
  min.cutoff = NA,
  max.cutoff = NA,
  ncol = NULL,
  combine = TRUE,
  pt.size.factor = 1.6,
  alpha = c(1, 1),
  image.alpha = 1,
  stroke = 0.25,
  interactive = FALSE,
  information = NULL,
  column.name = "all_IsomerNames",
  plot.exact = TRUE,
  verbose = TRUE
)
```

Arguments

object	A SpatialExperiment or aligned Cardinal MSImagingExperiment.
metabolites	Metabolite names to search in feature annotations.
plusminus	Optional non-negative absolute m/z tolerance.
images	Optional image IDs in imgData.
crop	Crop to the selected pixels.
assay	Primary experiment ("main") or altExp name.

slot	Expression assay name, e.g. "counts" or "logcounts". See experimentAccess for the experiment/matrix distinction.
keep.scale	Colour scaling: "feature", "all" or "none"; NULL means "none".
min.cutoff, max.cutoff	One cutoff or one per query; numeric or quantile strings such as "q10". NA uses the observed range.
ncol	Number of columns in combined plots.
combine	Combine panels; FALSE returns a list of ggplots.
pt.size.factor	Point size.
alpha	Point opacity, or a two-value range mapped to expression.
image.alpha	Optical image opacity.
stroke	Point border width.
interactive	Return Plotly widgets rather than static plots.
information	Deprecated Seurat hover data; non-NULL is unsupported.
column.name	Annotation column in rowData.
plot.exact	Match complete semicolon-delimited names (case-insensitive).
verbose	Retained for call compatibility.

Value

A ggplot or Plotly widget; a list when combine=FALSE.

Examples

```
utils::str(formals(spatialMZAnnotationPlot))
```

spatialMZPlot

Plot MSI features with optional optical images

Description

Images come from imgData and use its scaleFactor to place the raster in spatialCoords units. Each sample/image is a separate panel. With images=NULL all samples are plotted without an image.

Usage

```
spatialMZPlot(
  object,
  mzs,
  plusminus = NULL,
  images = NULL,
  crop = TRUE,
  assay = "main",
  slot = "counts",
```

```

    keep.scale = "feature",
    min.cutoff = NA,
    max.cutoff = NA,
    ncol = NULL,
    combine = TRUE,
    pt.size.factor = 1.6,
    alpha = c(1, 1),
    image.alpha = 1,
    stroke = 0.25,
    interactive = FALSE,
    information = NULL,
    verbose = TRUE
  )

```

Arguments

<code>object</code>	A <code>SpatialExperiment</code> or aligned <code>Cardinal MSImagingExperiment</code> .
<code>mzs</code>	Numeric <code>m/z</code> values or character feature IDs.
<code>plusminus</code>	Optional non-negative absolute <code>m/z</code> tolerance.
<code>images</code>	Optional image IDs in <code>imgData</code> .
<code>crop</code>	Crop to the selected pixels.
<code>assay</code>	Primary experiment ("main") or <code>altExp</code> name.
<code>slot</code>	Expression assay name, e.g. "counts" or "logcounts". See experimentAccess for the experiment/matrix distinction.
<code>keep.scale</code>	Colour scaling: "feature", "all" or "none"; NULL means "none".
<code>min.cutoff, max.cutoff</code>	One cutoff or one per query; numeric or quantile strings such as "q10". NA uses the observed range.
<code>ncol</code>	Number of columns in combined plots.
<code>combine</code>	Combine panels; FALSE returns a list of ggplots.
<code>pt.size.factor</code>	Point size.
<code>alpha</code>	Point opacity, or a two-value range mapped to expression.
<code>image.alpha</code>	Optical image opacity.
<code>stroke</code>	Point border width.
<code>interactive</code>	Return Plotly widgets rather than static plots.
<code>information</code>	Deprecated Seurat hover data; non-NULL is unsupported.
<code>verbose</code>	Retained for call compatibility.

Value

A ggplot or Plotly widget; a list when `combine=FALSE`.

Examples

```
utils::str(formals(spatialMZPlot))
```

spectralBinning	<i>Spectral binning of intensity values stored in a Matrix object, converted from matter.</i>
-----------------	---

Description

Spectral binning of intensity values stored in a Matrix object, converted from matter.

Usage

```
spectralBinning(  
  matrix,  
  ref,  
  index,  
  method = c("sum", "mean", "max", "min"),  
  tolerance  
)
```

Arguments

matrix	matter matrix object containing the intensity values to be binned.
ref	A vector of reference mass-to-charge (m/z) values for binning. If left unspecified, mass range will be used to generate reference peaks.
index	Character string specifying the column name for the m/z values (default = "mz").
method	A character vector specifying the binning method. Options include "sum", "mean", "max", "min". If not specified default method used is "sum".
tolerance	Numeric value specifying the tolerance for m/z matching (default = NA).

Value

Matrix object containing the binned intensity values matching the provided reference list.

Examples

```
utils::str(formals(spectralBinning))  
#Helper function for binning data in Matrix format
```

statPlot

*Helper function for QC plots by generating intensity count data***Description**

Helper function for QC plots by generating intensity count data

Usage

```
statPlot(
  data,
  group.by = NULL,
  assay = "main",
  slot = "counts",
  bottom.cutoff = NULL,
  top.cutoff = NULL,
  log.data = FALSE,
  verbose = FALSE
)
```

Arguments

data	A SingleCellExperiment, including SpatialExperiment.
group.by	Name of the colData() column to group by (default = NULL).
assay	Primary (main) or alternative experiment name.
slot	Expression assay name within the selected experiment. See experimentAccess for the experiment/matrix distinction.
bottom.cutoff	Numeric value defining the percent of data to exclude for the lower end of the distribution. A bottom.cutoff = 0.05 will remove the bottom 5% of data point (default = NULL).
top.cutoff	Numeric value defining the percent of data to exclude for the upper end of the distribution. A top.cutoff = 0.05 will remove the top 5% of data point (default = NULL).
log.data	Boolean value indicating whether to log transform the y-axis values (default = FALSE).
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = FALSE).

Value

A data.frame containing the relative transformed and sum counts required for various QC plots

Examples

```
utils::str(formals(statPlot))
```

subsetMZFeatures	<i>Subset a Bioconductor experiment by a list of m/z's</i>
------------------	--

Description

Subset a Bioconductor experiment by a list of m/z's

Usage

```
subsetMZFeatures(data, features, assay = "main")
```

Arguments

data	A Bioconductor experiment for subsetting.
features	A list of character strings defining the features/mz values to subset against.
assay	A character string identifying the primary or alternative experiment which contains the count data being subset.

Value

A subset Bioconductor experiment containing only m/z values that were specified

Examples

```
utils::str(formals(subsetMZFeatures))
```

subsetSPM	<i>Subset features and pixels through Bioconductor containers</i>
-----------	---

Description

Uses the container's standard bracket method so alternative experiments, reduced dimensions and spatial edges remain synchronized with pixels.

Usage

```
subsetSPM(
  object = NULL,
  subset = NULL,
  cells = NULL,
  idents = NULL,
  features = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

object	A SummarizedExperiment, including SingleCellExperiment or SpatialExperiment.
subset	Logical expression indicating features/variables to keep
cells	A vector of cells to keep; if NULL, defaults to all cells
idents	A vector of identity classes to keep
features	A vector of feature names or indices to keep
verbose	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = FALSE).
...	Reserved; additional arguments are rejected.

Details

Function params/args:

Value

A subset of the same Bioconductor container class.

Examples

```
utils::str(formals(subsetSPM))
# subsetSPM(spe, subset = region == "edge")
```

tmmNormalize	<i>Performs TMM normalization between categories based on a specified ident</i>
--------------	---

Description

This function is mainly used for normalising a merged Bioconductor experiment containing multiple samples.

Usage

```
tmmNormalize(
  combined.obj,
  ident,
  refIdent,
  normalisation.type = "CPM",
  CPM.scale.factor = 1e+06,
  assay = "main",
  slot = "counts",
  verbose = FALSE
)
```

Arguments

<code>combined.obj</code>	Bioconductor experiment that contains groups being normalized.
<code>ident</code>	Character string defining the column name or ident group to normalize between.
<code>refIdent</code>	Character string specifying one class/group type to use as a reference for TMM normalisation.
<code>normalisation.type</code>	Character string defining the normalization method to run. Options are either <code>c("CPM", "TIC", "LogNormalize")</code> which represent counts per million (CPM), Total Ion Current (TIC) normalization or Log Normalization, respectively (default = "CPM").
<code>CPM.scale.factor</code>	Numeric value that sets the scale factor for pixel/spot level normalization. Following normalization the total intensity value across each pixel will equal this value (default = 1e6).
<code>assay</code>	Primary (main) or alternative experiment name.
<code>slot</code>	Expression assay name within the selected experiment. See experimentAccess for the experiment/matrix distinction.
<code>verbose</code>	Boolean indicating whether to show the message. If TRUE the message will be show, else the message will be suppressed (default = FALSE).

Value

Bioconductor experiment with count values normalised and corrected for between categories

Examples

```
utils::str(formals(tmmNormalize))
```

<code>visualisePathways</code>	<i>Visualise Significant Pathways</i>
--------------------------------	---------------------------------------

Description

Displays the pathway analysis results form running the 'fishersPathwayAnalysis()' function

Usage

```
visualisePathways(
  SpaMTP,
  pathway_df,
  assay = "SPM",
  slot = "counts",
  min_n = 3,
  p_val_threshold = 0.1,
  method = "ward.D2",
```

```

    verbose = TRUE,
    database = NULL,
    database_version = "latest",
    database_source = c("auto", "spamtpdb"),
    database_local_dir = NULL,
    ...
)

```

Arguments

SpaMTP	A SpatialExperiment used for fishersPathwayAnalysis.
pathway_df	Dataframe containing the pathway enrichment results (output from SpaMTP::fishersPathwayAnalysis function).
assay	Primary ("main") or alternative experiment name. The default "SPM" is a compatibility alias for the primary experiment.
slot	Expression assay name within the selected experiment (default = "counts"), read with SummarizedExperiment::assay(). See experimentAccess for the experiment/matrix distinction.
min_n	Integer value specifying the minimum number of analytes required to be present in a pathway (default = 3).
p_val_threshold	The p-val cutoff to keep the pathways generated from fisher exact test (default = "0.1").
method	Character string defining the statistical method used to calculate hclust (default = "ward.D2").
verbose	Boolean indicating whether to show informative messages. If FALSE these messages will be suppressed (default = TRUE).
database	Optional named list of database resources, normally created by loadSpaMTPDatabase() .
database_version	SpaMTPdb/RaMP version used for pathway lookup.
database_source	Database source; see loadSpaMTPDatabase() .
database_local_dir	Optional staged SpaMTPdb resource directory.
...	The arguments pass to stats::hclust

Value

A combined gg, ggplot object with pathway and dendrogram

Examples

```
utils::str(formals(visualisePathways))
```

Index

* datasets

- adduct_file, [12](#)
- reaction_type, [121](#)
- add_backslashes_to_specialfeatures, [5](#)
- addCustomMZAnnotations, [5](#)
- addFMP10Annotations, [6](#)
- addSMImage, [8](#)
- addSpatialImage, [9](#)
- addSpatialImage(), [141](#)
- addSpatialImage, SpatialExperiment-method
(addSpatialImage), [9](#)
- addSSCAnnotation, [10](#)
- addTranscriptome, [11](#)
- addTranscriptome(), [59](#)
- addTranscriptome, SpatialExperiment-method
(addTranscriptome), [11](#)
- adduct_file, [12](#)
- adductRules, [12](#)
- adductRules(), [12](#), [35](#)
- alignSpatialOmics, [13](#)
- analyzeSpaMTPRegions, [14](#)
- annotateBigData, [15](#)
- annotateGeneIdentifiers, [17](#)
- annotateMZ, [18](#)
- annotateSM, [19](#)
- annotateSM(), [102](#)
- annotateSMILESStructure, [21](#)
- annotateTable, [22](#)
- annotationInfo, [24](#)
- applySpatialAlignment, [24](#)
- asCardinal, [28](#)
- asCardinal, MSImagingArrays-method
(asCardinal), [28](#)
- asCardinal, MSImagingExperiment-method
(asCardinal), [28](#)
- asCardinal, SpatialExperiment-method
(asCardinal), [28](#)
- asSpatialExperiment, [29](#)
- asSpatialExperiment(), [59](#)
- asSpatialExperiment, MSImagingArrays-method
(asSpatialExperiment), [29](#)
- asSpatialExperiment, MSImagingExperiment-method
(asSpatialExperiment), [29](#)
- asSpatialExperiment, SpatialExperiment-method
(asSpatialExperiment), [29](#)
- asSpatialExperiment, SummarizedExperiment-method
(asSpatialExperiment), [29](#)
- binMetabolites, [30](#)
- binnedCardinalToSeurat, [30](#)
- binSpaMTP, [31](#)
- binSpaMTP, ANY-method (binSpaMTP), [31](#)
- binSpaMTP, MSImagingArrays-method
(binSpaMTP), [31](#)
- binSpaMTP, MSImagingExperiment-method
(binSpaMTP), [31](#)
- binSpaMTP, SpatialExperiment-method
(binSpaMTP), [31](#)
- buildGeneMappingIndex, [33](#)
- buildGeneMappingIndex(), [36](#), [39](#), [49](#), [52](#),
[66](#), [69](#), [89](#), [104](#), [108](#), [110](#), [130](#)
- buildMZAnnotationIndex, [34](#)
- buildMZAnnotationIndex(), [19](#), [88](#),
[118–120](#)
- buildPathwayIndex, [35](#)
- calculate_bounds, [37](#)
- calculateAnnotationStatistics, [38](#)
- calculateSingleAnnotationStatistics,
[40](#)
- calculateSingleAnnotationStatistics(),
[39](#)
- cardinalToSeurat, [41](#)
- cardinalToSpatialExperiment, [42](#)
- check_and_truncate_adduct_vector, [43](#)
- check_cardinal_version, [44](#)
- checkAlignment, [44](#)
- combineTransformations, [45](#)
- convertSeuratToCardinal, [46](#)

- createMergedModalityAssay, [47](#)
- createPathwayAssay, [48](#)
- createPathwayAssay(), [102](#)
- createPathwayObject, [51](#)
- createPathwayObject(), [48](#)

- db_adduct_filter, [53](#)
- deconvolveSMILES, [54](#)
- deconvolveSMILES(), [21](#), [35](#), [118](#)
- demsHeatmap, [55](#)
- densityMap, [57](#)

- experimentAccess, [30](#), [33](#), [38](#), [40](#), [41](#), [46](#), [47](#),
[49](#), [51](#), [57](#), [58](#), [60](#), [62](#), [63](#), [65](#), [67](#), [78](#),
[80](#), [81](#), [90](#), [92](#), [96](#), [97](#), [99](#), [101–103](#),
[106](#), [107](#), [109](#), [113](#), [114](#), [127](#), [128](#),
[134](#), [136](#), [137](#), [145](#), [146](#), [148](#), [151](#),
[152](#)

- findAllDEMs, [59](#)
- findAllDEMs(), [125](#)
- findCorrelatedFeatures, [61](#)
- findDuplicateAnnotations, [63](#)
- findNearestMZ, [63](#)
- findRegionalPathways, [64](#)
- findRegionalPathways(), [102](#)
- findSpatiallyVariableMetabolites, [67](#)
- fishersPathwayAnalysis, [68](#)
- fishersPathwayAnalysis(), [66](#)
- formula_filter, [71](#)

- get_analytes_db, [71](#)
- getKmeanClusters, [72](#)
- getMetaspace, [73](#)
- getMZMetadata, [74](#)
- getRefinedAnnotations, [75](#)
- getSpatiallyVariableMetabolites, [75](#)
- getSquareCoordinates, [76](#)

- imageMZAnnotationPlot, [77](#)
- imageMZPlot, [79](#)
- interactiveSpatialPlot, [81](#)

- kNeighborsGraph, [82](#)

- labels_to_show, [83](#)
- loadMetaspace, [83](#)
- loadSM, [84](#)
- loadSpaMTPDatabase, [85](#)

- loadSpaMTPDatabase(), [7](#), [16](#), [19](#), [21](#), [23](#), [36](#),
[49](#), [52](#), [65](#), [66](#), [69](#), [103](#), [111](#), [112](#),
[130](#), [152](#)

- malDIMatrixProfiles, [87](#)
- malDIMatrixRules, [87](#)
- malDIMatrixRules(), [12](#), [35](#)
- mapGeneIdentifiers, [88](#)
- mapGeneIdentifiers(), [17](#)
- mapSpatialOmics, [90](#)
- massIntensityPlot, [91](#)
- metaspaceClient, [93](#)
- metaspaceToFeatureMatrix, [93](#)
- multiOmicIntegration, [94](#)
- multiOmicIntegration, ANY-method
(multiOmicIntegration), [94](#)
- multiOmicIntegration, SpatialExperiment-method
(multiOmicIntegration), [94](#)
- mzBoxPlot, [95](#)
- mzRidgePlot, [97](#)
- mzViolinPlot, [98](#)

- normalizeSMDData, [99](#)
- normalizeSMDData(), [59](#)
- normalizeSMDData, ANY-method
(normalizeSMDData), [99](#)
- normalizeSMDData, MSImagingArrays-method
(normalizeSMDData), [99](#)
- normalizeSMDData, MSImagingExperiment-method
(normalizeSMDData), [99](#)
- normalizeSMDData, SingleCellExperiment-method
(normalizeSMDData), [99](#)

- pathwayNetworkPlots, [101](#)
- pixelPlot, [105](#)
- plot3DFeature, [105](#)
- plotPathways, [107](#)
- plotPathwaysSpatially, [109](#)
- plotRegionalPathways, [111](#)
- plotSinglePathway, [112](#)
- plotSinglePathwaySpatially, [113](#)
- plotSpatialFeature, [115](#)
- plotSpatialFeature, MSImagingExperiment-method
(plotSpatialFeature), [115](#)
- plotSpatialFeature, SpatialExperiment-method
(plotSpatialFeature), [115](#)
- ppm_error, [117](#)
- ppm_range_match, [117](#)
- predictAdductsFromSMILES, [118](#)

`print.spamtp_mz_index`, 119
`Pseudo_msms`, 119

`queryMZAnnotationIndex`, 120
`queryMZAnnotationIndex()`, 19

`reaction_type`, 121
`readSMMatrix`, 122
`refineLipids`, 123
`renderSpaMTPReport`, 124
`runDE`, 125
`runMetabolicPCA`, 126
`runMetabolicPCA()`, 59
`runPooling`, 128
`runRAMPGeseca`, 129
`runSpaMTPWorkflow`, 131
`runSpatialGraphPCA`, 133

`savePheatmapAsPDF`, 135
`saveSpaMTPData`, 135
`scalesSMDData`, 137
`scalesSMDData()`, 59
`searchAnnotations`, 138
`selectROIs`, 139
`seuratToSingleCellExperiment`, 140
`seuratToSingleCellExperiment()`, 59, 141
`seuratToSpatialExperiment`, 141
`seuratToSpatialExperiment()`, 46, 59
`spaMTPDatabaseInfo`, 142
`spaMTPDatabaseInfo()`, 86
`SpaMTPdb`, 85
`SpaMTPdb::spaMTPdbGeneReference()`, 34, 36, 39, 49, 52, 66, 69, 89, 104, 108, 110, 130

`spatialExperimentToCardinal`, 142
`spatialExperimentToSeurat`, 143
`spatialExperimentToSeurat()`, 59
`spatialMZAnnotationPlot`, 144
`spatialMZPlot`, 145
`spectralBinning`, 147
`statPlot`, 148
`subsetMZFeatures`, 149
`subsetSPM`, 149

`tmmNormalize`, 150

`visualisePathways`, 151