Package: neuroSCC (via r-universe)

March 28, 2025

Type Package

Title Bridging Simultaneous Confidence Corridors and PET Neuroimaging

Version 1.0.0

Maintainer Juan A. Arias Lopez <juanantonio.arias.lopez@usc.es>

Description Tools for the structured processing of PET neuroimaging data in preparation for the estimation of Simultaneous Confidence Corridors (SCCs) for one-group, two-group, or single-patient vs group comparisons. The package facilitates PET image loading, data restructuring, integration into a Functional Data Analysis framework, contour extraction, identification of significant results, and performance evaluation. It bridges established packages (e.g., 'oro.nifti') with novel statistical methodologies (e.g., 'ImageSCC') and enables reproducible analysis pipelines, including comparison with Statistical Parametric Mapping ('SPM').

License MIT + file LICENSE

URL https://iguanamarina.github.io/neuroSCC/, https://github.com/iguanamarina/PhD-2023-SCC-vs-SPM-Group-vs-Group, https://github.com/iguanamarina/PhD-2024-SCC-vs-SPM-SinglePatient-vs-Group

BugReports https://github.com/iguanamarina/neuroSCC/issues

Depends R (>= 4.2.0)

Imports contoureR, dplyr, graphics, memisc, oro.nifti, stats, tidyr, utils

Suggests BPST, ImageSCC, knitr, rcmdcheck, remotes, rmarkdown, roxygen2 (>= 7.1.0), Triangulation

Additional_repositories https://iguanamarina.github.io/neuroSCC-drat

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Config/pak/sysreqs libicu-dev

Repository https://iguanamarina.r-universe.dev

RemoteUrl https://github.com/iguanamarina/neuroscc

RemoteRef HEAD

RemoteSha 0a323dc86cd21f0c6ed3844235515b21bbfb3b72

Contents

`neuroSCC-package` *Wrappers for Neuroimaging Functional Data Preparation and SCC Analysis*

Description

The `neuroSCC` package provides tools to preprocess and structure neuroimaging data for functional data analysis using Simultaneous Confidence Corridors (SCCs). It wraps external packages to prepare data from PET images, extract contours, generate meshes, and evaluate regions of statistical significance.

The methods implemented support both group comparisons and single-subject vs. group inference, following the methodology described in Wang et al. (2020) and the author's PhD thesis.

Details

This package serves as a bridge between neuroimaging file formats (e.g., NIfTI) and advanced statistical tools like `ImageSCC::scc.image`. It includes the following key components.

- Loading and cleaning PET image data.
- Extracting ROIs and constructing functional data matrices.

- Generating synthetic Poisson clones for 1-vs-group settings.
- Extracting SCC-detected points and evaluating detection metrics.

Author(s)

Maintainer: Juan A. Arias Lopez <juanantonio.arias.lopez@usc.es> (ORCID) [copyright holder]

Other contributors:

- Virgilio Gomez Rubio <Virgilio.Gomez@uclm.es> (ORCID) [reviewer]
- Pablo Aguiar Fernandez <pablo.aguiar@usc.es> (ORCID) [thesis advisor]
- Andrew Haddon Kemp <A.H.Kemp@swansea.ac.uk> (ORCID) [thesis advisor]

See Also

neuroCleaner, databaseCreator, getPoints

---

calculateMetrics *Evaluate SCC or SPM Detection Performance*

Description

Computes Sensitivity, Specificity, Positive Predictive Value (PPV), and Negative Predictive Value (NPV) by comparing detected points with ground truth ROI points. This function is used to assess the accuracy of SCC- or SPM-based detection in neuroimaging analysis.

Usage

```
calculateMetrics(detectedPoints, truePoints, totalCoords, regionName)
```

Arguments

detectedPoints	A data frame containing detected coordinates (x, y). SCC-detected points should be obtained using getPoints. SPM-detected points should be obtained using getSPMbinary.
truePoints	A data frame with ground truth ROI coordinates (x, y), extracted via processROIs.
totalCoords	A list with the full voxel grid dimensions, created by getDimensions. Must include named elements xDim and yDim.
regionName	A character string used to label the output region.

Details

This function requires three precomputed objects

- detectedPoints: SCC-detected points from getPoints or SPM-detected points from getSPMbinary.
- truePoints: Ground truth ROIs extracted using processROIs.
- totalCoords: Full voxel coordinate grid from getDimensions.

Value

A data frame with the following evaluation metrics

- `region`: Name of the analyzed region.
- `sensitivity`: True positive rate (TP / (TP + FN) * 100).
- `specificity`: True negative rate (TN / (TN + FP) * 100).
- `PPV`: Positive predictive value (TP / (TP + FP) * 100).
- `NPV`: Negative predictive value (TN / (TN + FN) * 100).

See Also

`getPoints` for SCC-detected regions.
`getSPMbinary` for binary SPM-detected points.
`processROIs` for defining ground truth ROIs.
`getDimensions` for generating the coordinate grid.

Examples

```
# Load precomputed inputs for the example
data("calculateMetricsExample", package = "neuroSCC")

# Evaluate SCC and SPM detection performance
with(calculateMetricsExample, {
  metricsSCC <- calculateMetrics(detectedSCC, trueROI, totalCoords, "Region2_SCC")
  metricsSPM <- calculateMetrics(detectedSPM, trueROI, totalCoords, "Region2_SPM")

  print(metricsSCC)
  print(metricsSPM)
})
```

---

calculateMetricsExample

Precomputed Inputs for SCC vs. SPM Performance Evaluation

Description

A dataset containing all necessary inputs for demonstrating `calculateMetrics`. It enables reproducible and fast example code that compares SCC-detected and SPM-detected points against a known ground truth ROI.

These inputs were generated using sample PET and ROI files included in the `neuroSCC` package.

Usage

```
data("calculateMetricsExample")
```

Format

A set of four objects:

`detectedSCC` Data frame of SCC-detected coordinates (from `getPoints`).

`detectedSPM` Data frame of SPM-detected coordinates (from `getSPMbinary`).

`trueROI` Ground truth ROI voxel data (from `processROIs`).

`totalCoords` List with full image grid dimensions (from `getDimensions`).

Source

Simulated PET neuroimaging study for testing SCC and SPM detection accuracy.

See Also

`calculateMetrics`, `getPoints`, `getSPMbinary`, `processROIs`, `getDimensions`

`databaseCreator` *Create a Database of Processed PET Image Data*

Description

Processes multiple PET image files matching a specified filename pattern. Each file is processed using `neuroCleaner`, and the results are aggregated into a unified data frame for functional data analysis. This function serves as a key step in the `neuroSCC` workflow, bridging raw image data and Simultaneous Confidence Corridors (SCC) computation.

Usage

```
databaseCreator(
  pattern,
  control = TRUE,
  useSequentialNumbering = FALSE,
  demo = NULL,
  quiet = FALSE
)
```

Arguments

`pattern`	`character`. A regular expression defining the file pattern to match. Subject identifiers are extracted from filenames based on this pattern.
`control`	`logical`. If `TRUE`, files are treated as control group data; if `FALSE`, as pathological group data. Default is `TRUE`.
`useSequentialNumbering`	`logical`. If `TRUE`, assigns sequential subject numbers instead of extracting them from filenames. Default is `FALSE`.
`demo`	`data.frame`, optional. If provided, demographic information is included for each file. Default is `NULL`.
`quiet`	`logical`. If `TRUE`, suppresses progress messages. Default is `FALSE`.

Details

The function performs the following steps

1. Identifies image files matching the given pattern.
2. Processes each file using `neuroCleaner`, optionally merging demographic data.
3. Adds a subject identifier column (`CN_number` or `AD_number`).
4. Aggregates all results into a single data frame.

If no files are successfully processed, an empty data frame is returned with a warning.

This function is typically followed by `matrixCreator`, which converts the output into a matrix format for functional analysis.

Value

A `data.frame` combining processed voxel-level data from all matched files. Each row represents a voxel (3D pixel). The column structure depends on input

- For the control group: `CN_number`, `z`, `x`, `y`, `pet`
- For the pathological group: `AD_number`, `z`, `x`, `y`, `pet`
- If demographics are included: additional columns `PPT`, `Group`, `Sex`, `Age`

See Also

`neuroCleaner` for the underlying image processing function.
`matrixCreator` for the next step in the workflow that converts the database to a matrix format for SCC analysis.

Examples

```
# NOTE: To keep runtime below CRAN limits, this example processes only 1 subject.
# You can expand the pattern to include all subjects for real use.

# Example: Create a database from a single synthetic PET image (control group)
controlPattern <- "^syntheticControl1\\.nii\\.gz$"
databaseControls <- databaseCreator(pattern = controlPattern, control = TRUE, quiet = TRUE)
head(databaseControls)
```

`generatePoissonClones` *Generate Synthetic Poisson Clones for PET Data*

Description

Generates synthetic clones of a PET data matrix by adding Poisson-distributed noise to each non-zero voxel. This approach helps address the limitations of functional data analysis (FDA) in single-subject versus group (1 vs. Group) setups, where a single subject lacks sufficient variability to reliably estimate Simultaneous Confidence Corridors (SCCs).

Usage

```
generatePoissonClones(originalMatrix, numClones, lambdaFactor)
```

Arguments

originalMatrix	A numeric matrix where each row represents a flattened PET image.
numClones	An integer specifying the number of synthetic clones to generate.
lambdaFactor	A positive numeric value that scales the magnitude of Poisson noise.

Details

- Values equal to 0 remain unchanged to preserve background regions.
- NA values are replaced with 0 before adding noise.
- Poisson noise is applied only to positive values, scaled by lambdaFactor.
- Enables valid SCC estimation in single-subject settings by artificially increasing sample size.

Value

A numeric matrix with numClones rows, each representing a noisy version of originalMatrix with Poisson noise added.

Examples

```
# Load example input matrix for Poisson cloning
data("generatePoissonClonesExample", package = "neuroSCC")
# Select 10 random voxel positions for display
set.seed(123)
sampledCols <- sample(ncol(generatePoissonClonesExample), 10)
# Generate 1 synthetic clone
clones <- generatePoissonClones(generatePoissonClonesExample, numClones = 1, lambdaFactor = 0.25)
# Show voxel intensity values after cloning
clones[, sampledCols]
```

---

generatePoissonClonesExample

Example Input for Poisson Clone Generation

Description

A full single-subject PET matrix used to demonstrate generatePoissonClones. This matrix was extracted from simulated neuroimaging data included in the neuroSCC package.

The example avoids long runtime by generating only one synthetic clone.

Usage

```
data("generatePoissonClonesExample")
```

Format

A numeric matrix named `generatePoissonClonesExample`, with 1 row and all voxel columns.

Source

Simulated PET neuroimaging dataset included with `neuroSCC`.

See Also

`generatePoissonClones`

`getDimensions` *Get Dimensions from a Neuroimaging File*

Description

Extracts voxel dimension information from a NIfTI or similar neuroimaging file. This function is designed to work with `neuroCleaner`, but it can also be used independently to inspect image dimensions.

Usage

```
getDimensions(file)
```

Arguments

`file`	A NIfTI file object or a file path pointing to a NIfTI image.

Details

The function accepts either a file path or a preloaded `nifti` object. If a file path is provided, it uses `oro.nifti::readNIfTI()` to load the image. This function ensures consistent dimension extraction across the `neuroSCC` pipeline.

Value

A named list with the following elements

- `xDim` – Number of voxels along the X axis.
- `yDim` – Number of voxels along the Y axis.
- `zDim` – Number of slices along the Z axis.
- `dim` – Total number of voxels in a 2D slice (calculated as `xDim * yDim`).

Examples

```
# Get the file path for a sample NIfTI file
niftiFile <- system.file("extdata", "syntheticControl1.nii.gz", package = "neuroSCC")

# Extract dimensions from the NIfTI file
dimensions <- getDimensions(niftiFile)

# Display the extracted dimensions
print(dimensions)
```

`getPoints` *Extract Significant SCC Points from an SCC Comparison Object*

Description

Identifies and extracts coordinates where differences fall outside the simultaneous confidence corridors (SCCs), indicating statistically significant regions. This function processes the results from `ImageSCC::scc.image()` and returns the voxel locations that represent either hypo- or hyperactivity.

Interpretation depends on the order of inputs in the SCC computation. If SCC was computed as `scc.image(Ya = Y_AD, Yb = Y_CN, ...)` (i.e., the Control group is the second argument).

- `positivePoints` — Regions where Control minus Pathological is significantly above the SCC. These correspond to areas where the Pathological group (AD) is *hypoactive* relative to Controls.
- `negativePoints` — Regions where Control minus Pathological is significantly below the SCC. These correspond to areas where the Pathological group is *hyperactive* relative to Controls.

Always confirm the order of `Ya` **and** `Yb` **in the SCC computation** to interpret the directionality correctly.

Usage

```
getPoints(sccResult)
```

Arguments

`sccResult`	A list of SCC computation results, as returned by `ImageSCC::scc.image`. Must include the following components • `Z.band` — A matrix specifying grid positions. • `ind.inside.cover` — Indices of grid points within the confidence band. • `scc` — A 3D array containing the computed SCC values.

Value

A named list with two elements

- `positivePoints` — A data frame with coordinates where the **first group (Ya)** shows significantly lower activity than the **second group (Yb)**.
- `negativePoints` — A data frame with coordinates where the **first group (Ya)** shows significantly higher activity than the **second group (Yb)**.

See Also

`ImageSCC::scc.image` for SCC computation.

Examples

```
# Load precomputed SCC example
data("SCCcomp", package = "neuroSCC")

# Extract significant SCC points
significantPoints <- getPoints(SCCcomp)

# Show extracted points (interpretation depends on SCC setup; see description)
head(significantPoints$positivePoints)  # Pathological hypoactive vs. Control
head(significantPoints$negativePoints)  # Pathological hyperactive vs. Control
```

`getSPMbinary` *Extract SPM-Detected Significant Points from a Binary NIfTI File*

Description

Extracts voxel coordinates where `pet = 1` (i.e., statistically significant points) from a binary NIfTI file produced by an external SPM analysis. Only voxels from a specific brain slice (`z = paramZ`) are retained.

The output data frame is structured identically to that of `getPoints`, allowing direct comparison between SCC- and SPM-detected regions via `calculateMetrics`.

Usage

```
getSPMbinary(niftiFile, paramZ = 35)
```

Arguments

`niftiFile`	`character`. The path to the binary NIfTI file generated by SPM.
`paramZ`	`integer`. The specific z-slice to extract. Default is `35`.

Details

This function converts externally generated SPM results into a format compatible with SCC analysis tools in `neuroSCC`. Use `getDimensions` to inspect the full coordinate space if needed.

Value

A data frame with the following columns:

- x, y – Coordinates of significant voxels at the specified slice.

See Also

`getPoints` for SCC-based detection.
`getDimensions` for obtaining full coordinate grids.
`calculateMetrics` for evaluating SCC vs. SPM detection performance.

Examples

```
# Load a sample binary NIfTI file (SPM result)
niftiFile <- system.file("extdata", "binary.nii.gz", package = "neuroSCC")
detectedSPM <- getSPMbinary(niftiFile, paramZ = 35)

# Show detected points
head(detectedSPM)
```

---

matrixCreator *Convert Database to Functional Data Matrix Format*

Description

Converts a PET image database (created via `databaseCreator`) into a matrix format suitable for functional data analysis. Each row of the resulting matrix corresponds to a subject, and each column to a voxel's PET intensity values at a specified brain slice.

Usage

```
matrixCreator(
  database,
  paramZ = 35,
  useSequentialNumbering = FALSE,
  quiet = FALSE
)
```

Arguments

`database`	A data frame created by `databaseCreator`, containing voxel-level PET image data, including subject identifiers, coordinates, and intensity values.
`paramZ`	An integer specifying the z-coordinate (slice) to extract. Default is `35`.
`useSequentialNumbering`	`logical`. If `TRUE`, assigns sequential subject IDs instead of extracting them from filenames. Not currently used inside this function. Default is `FALSE`.
`quiet`	`logical`. If `TRUE`, suppresses progress messages. Default is `FALSE`.

Details

This function performs the following steps

1. Verifies that the specified z-slice exists in the database.
2. Identifies the correct subject grouping column (`CN_number` or `AD_number`).
3. Determines the matrix dimensions using `x` and `y` coordinates.
4. Extracts PET intensities per subject at the given slice.
5. Replaces any `NaN` values with `0` to ensure numerical stability.

This function typically follows `databaseCreator` and precedes `meanNormalization` in the neuroSCC workflow.

Value

A numeric matrix where

- Each row represents one subject's PET values at the selected z-slice.
- Each column corresponds to a voxel (flattened as a 1D row).

See Also

`databaseCreator` for generating the input database.
`meanNormalization` for scaling matrix data prior to SCC computation.

Examples

```
# NOTE: To keep example runtime short, only one synthetic PET file is used.
# For full analysis, expand the filename pattern accordingly.

# Step 1: Generate a database for a single subject
controlPattern <- "^syntheticControl1\\.nii\\.gz$"
databaseControls <- databaseCreator(pattern = controlPattern, control = TRUE, quiet = TRUE)

# Step 2: Convert the database into a matrix format
matrixControls <- matrixCreator(databaseControls, paramZ = 35, quiet = TRUE)

# Display dimensions of the matrix
dim(matrixControls)
```

meanNormalization *Mean Normalization for Matrix Data*

Description

Normalizes each row of a matrix by dividing its elements by the row mean, ignoring NA values. This step is commonly used to adjust for global intensity differences across subjects before applying statistical comparisons or functional data analysis.

Usage

```
meanNormalization(
  matrixData,
  handleInvalidRows = c("warn", "error", "omit"),
  returnDetails = FALSE,
  quiet = FALSE
)
```

Arguments

matrixData	A matrix where each row represents one subject's PET data, typically generated by matrixCreator.
handleInvalidRows	character. Specifies how to handle rows with invalid means (either zero or NA). Options include "warn" (default), "error", or "omit".
returnDetails	logical. If TRUE, returns a list with the normalized matrix and additional diagnostics. If FALSE (default), returns only the normalized matrix.
quiet	logical. If TRUE, suppresses console messages. Default is FALSE.

Details

The function performs the following steps

1. Computes the row means of the input matrix, ignoring NAs.
2. Divides each row by its corresponding mean.
3. Replaces NaN values (from division by 0) with 0 if applicable.
4. Handles problematic rows according to the selected handleInvalidRows option: "warn" (default) issues a warning, "error" stops execution, and "omit" removes the affected rows from the result.

This step is often used prior to applying SCC methods to ensure comparability across subjects.

Value

A normalized matrix, or a list if returnDetails = TRUE.

- normalizedMatrix – The normalized matrix.
- problemRows – Indices of rows that had zero or NA means.

See Also

matrixCreator for building the matrix input to normalize.

Examples

```
# Generate a minimal database and create a matrix (1 control subject)
dataDir <- system.file("extdata", package = "neuroSCC")
controlPattern <- "^syntheticControl1\\.nii\\.gz$"
databaseControls <- databaseCreator(pattern = controlPattern,
                                    control = TRUE,
                                    quiet = TRUE)
matrixControls <- matrixCreator(databaseControls, paramZ = 35, quiet = TRUE)

# Normalize the matrix (with diagnostics)
normalizationResult <- meanNormalization(matrixControls,
                                         returnDetails = TRUE,
                                         quiet = FALSE)
```

---

neuroCleaner *Clean and Load Data from NIfTI Neuroimaging Files*

Description

Loads a NIfTI-format neuroimaging file and transforms it into a structured data frame, organizing voxel-level information for downstream analysis. This function is the first step in the neuroimaging processing pipeline in neuroSCC, converting raw PET data into a format suitable for functional data analysis. SCCs are later computed using functions from the ImageSCC package, such as ImageSCC::scc.image().

Usage

```
neuroCleaner(name, demo = NULL, demoRow = 1)
```

Arguments

name	character. The full path to the NIfTI file to process.
demo	Optional data.frame containing demographic information. If provided, it should include columns (case-insensitive): PPT, Group, Sex, and Age. If automatic matching via the PPT column fails, the row specified by demoRow is used. Default is NULL.
demoRow	integer. Row to use from the demographic table if automatic matching fails. Default is 1.

Details

The function performs the following steps

1. Loads the NIfTI file using `oro.nifti::readNIfTI()`.
2. Converts the 3D image into a tidy data frame.
3. Adds `z`, `x`, and `y` voxel coordinates.
4. If demographic data is provided, attempts to match based on `PPT` (case-insensitive). If no match is found, `demoRow` is used.

The resulting data frame serves as input for `databaseCreator`, `matrixCreator`, and other core functions in the `neuroSCC` pipeline.

Value

A data frame where each row represents a voxel (3D pixel).

- If demographics are provided: the columns include `PPT`, `Group`, `Sex`, `Age`, `z`, `x`, `y`, and `pet`.
- If demographics are not provided: the columns include `z`, `x`, `y`, and `pet`.

The `pet` column contains the PET intensity value at each voxel location.

See Also

`databaseCreator` for batch image processing.
`readNIfTI` for reading NIfTI-format files.

Examples

```
# Load a sample Control NIfTI file
niftiFile <- system.file("extdata", "syntheticControl1.nii.gz", package = "neuroSCC")

# Example Without demographic data
petData <- neuroCleaner(niftiFile)
petData[sample(nrow(petData), 10), ]  # Show 10 random voxels
```

`neuroContour` *Obtain and save neuroimaging contours from a NIFTI file*

Description

This function extracts contours from a neuroimaging NIFTI file where values change according to specified levels. It processes the NIFTI file with `neuroCleaner` to extract structured neuroimaging data, then extracts contours using `contoureR::getContourLines`. These contours serve as input for `Triangulation::TriMesh`, which is used in Simultaneous Confidence Corridors (SCCs) calculations.

While **not mandatory**, it is **highly recommended** that the input NIFTI file be pre-processed such that zero values represent the background and non-zero values represent regions of interest. The function's default behavior extracts contours at level `0`, which is ideal for well-masked data.

Usage

```
neuroContour(niftiFile, paramZ = 35, levels = c(0), plotResult = FALSE)
```

Arguments

`niftiFile`	`character`, the path to the NIFTI file containing neuroimaging data. Ideally, the file should be masked so that zero values represent the background.
`paramZ`	`integer`, the specific z-slice to extract contours from. Default is `35`.
`levels`	`numeric`, a vector of levels at which to draw the contours. Default is `c(0)`.
`plotResult`	`logical`, if `TRUE`, plots the extracted contours. Default is `FALSE`.

Details

This function extracts contours from a **NIFTI** file, typically a **masked** image where background values are set to zero, and regions of interest contain non-zero values. While users can specify a different boundary level, the recommended approach is to use `levels = 0` for masked data.

The extracted contours are typically used as input to `Triangulation::TriMesh` to create a triangular mesh of the region, which is then used for Simultaneous Confidence Corridors calculations.

Value

A `list` of data frames, where each data frame contains the x and y coordinates of a contour. The first element typically represents the external boundary, while subsequent elements (if present) represent internal contours or holes. Each data frame has two columns:

- `x` – x-coordinates of the contour points.
- `y` – y-coordinates of the contour points.

See Also

getContourLines for the underlying contour extraction.
`Triangulation::TriMesh` for the next step in the SCC calculation process.

Examples

```
# Get the file path for a sample NIfTI file
niftiFile <- system.file("extdata", "syntheticControl1.nii.gz", package = "neuroSCC")

# Extract contours at level 0
contours <- neuroContour(niftiFile, paramZ = 35, levels = 0, plotResult = TRUE)

# Display the first few points of the main contour
head(contours[[1]])
```

processROIs *Process ROI Voxel Data from a NIfTI File*

Description

Processes Regions of Interest (ROIs) from a binary NIfTI file by extracting voxel-level coordinates and labeling each voxel as part of the ROI or not. The function preserves the spatial structure and is typically used to prepare ground truth ROIs for comparison with SCC-detected regions via `calculateMetrics`.

Usage

```
processROIs(
  roiFile,
  region,
  number,
  save = TRUE,
  outputDir = tempdir(),
  verbose = TRUE
)
```

Arguments

`roiFile`	`character`. Path to the binary NIfTI file containing ROI data.
`region`	`character`. Name of the ROI region (e.g., `"Region2"`).
`number`	`character`. Identifier for the subject or group (e.g., `"18"`).
`save`	`logical`. If `TRUE`, saves the result as an `.RDS` file. If `FALSE`, returns a data frame in the console. Default is `TRUE`.
`outputDir`	`character`. Directory where the ROI table will be saved if `save = TRUE`. Default is a temporary file: `tempdir()`.
`verbose`	`logical`. If `TRUE`, displays progress messages. Default is `TRUE`.

Details

The function uses `neuroCleaner` to load and flatten the NIfTI file into a structured data frame. All voxels are retained, with the `pet` column indicating which ones are part of the ROI (`1`) versus background (`0`). An ROI label is added in the `group` column.

This output is used as ground truth for evaluating detection performance in SCC analyses.

Value

A data frame with voxel-level ROI information.

- `group` – Combined identifier built from `region` and `number`.
- `z`, `x`, `y` – Voxel coordinates.
- `pet` – Binary value indicating ROI membership (`1` = ROI, `0` = non-ROI).

If `save = TRUE`, the data frame is saved as an `.RDS` file and not returned to the console.

See Also

calculateMetrics for evaluating SCC detection performance.
neuroCleaner for reading and structuring voxel data.

Examples

```
# Load and process a sample ROI NIfTI file (console output)
```

SCCcomp *Example SCC Computation Result*

Description

A precomputed example of a Simultaneous Confidence Corridor (SCC) analysis comparing a group of pathological subjects against controls. This object was generated using the `ImageSCC::scc.image` function and represents a realistic output from SCC-based neuroimaging group comparisons.

This dataset is used in the examples of `getPoints` and `calculateMetrics`, allowing users to explore SCC outputs without needing to recompute them.

Usage

```
data("SCCcomp")
```

Format

A named list of class `"image"` with the following elements

`scc` 3D array of SCC confidence bands, dimensions `[n, 2, alpha]`.

`Z.band` Matrix of grid coordinates corresponding to evaluated locations.

`ind.inside.cover` Integer vector of indices for grid points inside the SCC band.

`V.est.a`, `V.est.b` Vertex matrices for triangulated domains (pathological and control groups).

`Tr.est.a`, `Tr.est.b` Triangle index matrices corresponding to the domain meshes.

`alpha` Vector of confidence levels used (e.g., 0.1, 0.05, 0.01).

`d.est` Spline degree used in mean function estimation.

`r` Smoothing parameter used during fitting.

Source

Simulated PET neuroimaging study for evaluating SCC methodology.

See Also

`getPoints`, `calculateMetrics`, `ImageSCC::scc.image`

Index