Package 'neuroSCC'

March 24, 2025

Type Package

Title Bridging Simultaneous Confidence Corridors with Applications and Performance Evaluation in PET Neuroimaging

Version 0.16.0

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

Description Provides functions 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, post-SCC identification of significant results, and performance evaluation. It bridges established neuroimaging packages (e.g., 'oro.nifti') with novel statistical methodologies (i.e., 'ImageSCC') and facilitates reproducible analysis pipelines, enabling performance comparison of SCC-based analyses with gold-standard methods such as Statistical Parametric Mapping (SPM).

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

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Depends R (>= 4.2.0)

Imports oro.nifti, memisc, contoureR, dplyr, tidyr

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

VignetteBuilder knitr

Funding This work was supported by an internship grant awarded at the 6th Conference of the Spanish National Biostatistics Network (BIOSTATNET) in 2025 as a prize for the best poster presentation and young researcher career.

License MIT + file LICENSE

NeedsCompilation no

Author Juan A. Arias Lopez [aut, cre],
Virgilio Gomez Rubio [rev],
Pablo Aguiar Fernandez [ths, cph],
Andrew Haddon Kemp [ths]

Contents

`neuroSCC-package` *neuroSCC: Bridging Simultaneous Confidence Corridors with Applications and Performance Evaluation in PET Neuroimaging*

Description

Provides functions 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, post-SCC identification of significant results, and performance evaluation. It bridges established neuroimaging packages (e.g., 'oro.nifti') with novel statistical methodologies (i.e., 'ImageSCC') and facilitates reproducible analysis pipelines, enabling performance comparison of SCC-based analyses with gold-standard methods such as Statistical Parametric Mapping (SPM).

Author(s)

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

Other contributors:

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

See Also

Useful links:

- 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
- Report bugs at https://github.com/iguanamarina/neuroSCC/issues

---

`calculateMetrics` *Evaluate SCC or SPM Detection Performance*

Description

Computes Sensitivity, Specificity, Positive Predictive Value (PPV), and Negative Predictive Value (NPV) for detected points compared to ground truth ROI points. This function is used to evaluate SCC-based and SPM-based detection accuracy in neuroimaging analysis.

Usage

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

Arguments

`detectedPoints`	A data frame containing SCC- or SPM-detected points (`x`, `y`). • SCC-detected points should come from `getPoints`. • SPM-detected points should come from `getSPMbinary`.
`truePoints`	A data frame containing ground truth ROI points (`x`, `y`), obtained using `processROIs`.
`totalCoords`	A data frame containing all possible voxel coordinates (`x`, `y`), obtained using `getDimensions`.
`regionName`	A character string used for labeling the results.

Details

The user must precompute the following objects before calling this function:

- `detectedPoints`: Extracted using `getPoints` (for SCC) or `getSPMbinary` (for SPM).
- `truePoints`: Extracted using `processROIs`, representing ground truth ROIs.
- `totalCoords`: Generated using `getDimensions`, providing the full voxel grid.

Value

A data frame with the following evaluation metrics:

- `region`: 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-based detection points. getSPMbinary for extracting SPM-detected points. processROIs for ground truth ROI extraction. getDimensions for obtaining the full coordinate grid.

Examples

```
# Extract detected SCC points
detectedSCC <- getPoints(SCCcomp)$positivePoints

# Extract detected SPM points
spmFile <- system.file("extdata", "binary.nii.gz", package = "neuroSCC")
detectedSPM <- getSPMbinary(spmFile, paramZ = 35)

# Extract true ROI points
roiFile <- system.file("extdata", "ROIsample_Region2_18.nii.gz", package = "neuroSCC")
trueROI <- processROIs(roiFile, region = "Region2", number = "18", save = FALSE)

# Generate totalCoords from getDimensions()
totalCoords <- getDimensions(roiFile)

# Compute SCC detection performance
metricsSCC <- calculateMetrics(detectedSCC, trueROI, totalCoords, "Region2_SCC")

# Compute SPM detection performance
metricsSPM <- calculateMetrics(detectedSPM, trueROI, totalCoords, "Region2_SPM")

# Print both results
print(metricsSCC)
print(metricsSPM)
```

---

databaseCreator *Create a database of processed PET image data*

Description

This function processes PET images based on a specified file name pattern. It reads each file matching the pattern, processes it using neuroCleaner, and compiles the results into a structured data frame. The function is a key step in the neuroSCC workflow, bridging individual image processing and preparation for functional data analysis with Simultaneous Confidence Corridors (SCCs).

Usage

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

Arguments

pattern	character, a regular expression pattern specifying which files to process. The function extracts the subject number directly from the filename based on this pattern.
control	logical, if TRUE, the function processes control group images; if FALSE, it processes pathological group images. Default is TRUE.
useSequentialNumbering	logical, if TRUE, assigns sequential numbers (1,2,3,...) to files instead of extracting from filenames. Default is FALSE.
demo	data.frame, optional demographic data. If provided, this information will be included in the output database for each file. Default is NULL.
quiet	logical, if TRUE, suppresses progress messages. Default is FALSE.

Details

The function performs the following operations:

1. Identifies files in the working directory that match the specified pattern.
2. For each file:
   - Extracts the subject number from the filename (or assigns a sequential number if useSequentialNumbering=TI
   - Processes the file using neuroCleaner, integrating demographic data if provided.
   - Adds subject identifier information to each row.
   - Merges all processed data into a structured database.

This function is typically followed by matrixCreator in the analysis pipeline, which transforms the database into a format suitable for SCC computation.

Value

A data.frame that aggregates processed data from all matched images. Each row represents data from one voxel (3D pixel), including:

- For control group (control=TRUE): CN_number, z, x, y, pet
- For pathological group (control=FALSE): AD_number, z, x, y, pet
- If demographic data is provided: Additional columns like PPT, Group, Sex, Age

The CN_number or AD_number column contains the subject identifier extracted from the filename or assigned sequentially. The pet column contains intensity values.

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

```
# Get the file path for sample data
dataDir <- system.file("extdata", package = "neuroSCC")

# Example 1: Create database for Controls
controlPattern <- "^syntheticControl.*\\.nii.gz$"
databaseControls <- databaseCreator(pattern = controlPattern, control = TRUE, quiet = TRUE)
head(databaseControls); tail(databaseControls)
nrow(databaseControls)  # Total number of rows
unique(databaseControls$CN_number)  # Show unique subjects

# Example 2: Create database for Pathological group
pathologicalPattern <- "^syntheticPathological.*\\.nii.gz$"
databasePathological <- databaseCreator(pattern = pathologicalPattern,
                                        control = FALSE,
                                        quiet = TRUE)
head(databasePathological); tail(databasePathological)
nrow(databasePathological)  # Total number of rows
unique(databasePathological$AD_number)  # Show unique subjects
```

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

Description

This function generates synthetic clones of a PET data matrix by adding Poisson-distributed noise to each non-zero voxel. This is necessary to circumvent the limitations of functional data analysis (FDA) in **single patient vs. group (1 vs. Group) setups**, where a single subject does not provide enough variability to estimate Simultaneous Confidence Corridors (SCCs) reliably.

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 controlling the magnitude of Poisson noise.

Details

- Values of '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'. - This approach allows **SCC methods to operate in 1 vs. Group analyses**, ensuring statistical validity when a single observation is not sufficient.

Value

A numeric matrix with 'numClones' rows, each containing a modified version of 'originalMatrix' with added Poisson noise.

Examples

```
# Get a single patient's PET data matrix
dataDir <- system.file("extdata", package = "neuroSCC")
pathologicalPattern <- "^syntheticPathological.*\\.nii.gz$"
databasePathological <- databaseCreator(pattern = pathologicalPattern,
                                        control = FALSE,
                                        quiet = TRUE)
matrixPathological <- matrixCreator(databasePathological, paramZ = 35, quiet = TRUE)
patientMatrix <- matrixPathological[1, , drop = FALSE]  # Select a single patient

# Select 10 random columns for visualization
set.seed(123)
sampledCols <- sample(ncol(patientMatrix), 10)

# Show voxel intensity values before cloning
patientMatrix[, sampledCols]

# Generate 5 synthetic clones with Poisson noise
clones <- generatePoissonClones(patientMatrix, numClones = 5, lambdaFactor = 0.25)

# Show voxel intensity values after cloning
clones[, sampledCols]
```

---

getDimensions *Get Dimensions from a Neuroimaging File*

Description

Extracts dimensional information from NIFTI or similar neuroimaging files. This function is designed to work together with `neuroCleaner()` but can also be used independently for informative purposes.

Usage

```
getDimensions(file)
```

Arguments

`file`	A NIFTI file object or filename to extract dimensions from.

Details

The function can handle both NIFTI file paths and pre-loaded oro.nifti file objects. It provides a consistent way to extract dimensional information across the package.

Value

A list containing:

- `xDim`: Number of voxels in the X dimension
- `yDim`: Number of voxels in the Y dimension
- `zDim`: Number of slices in the Z dimension
- `dim`: Total number of voxels (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

This function identifies and extracts coordinates where significant differences fall outside the simultaneous confidence corridors (SCCs). It processes the results from `ImageSCC::scc.image()`, returning the extracted coordinates where the differences are statistically significant.

The interpretation of the results depends on how the SCC was computed. If SCC was computed as `scc.image(Ya = Y_AD, Yb = Y_CN, ...)`, meaning the **Control group (CN) is the second argument**, the function extracts:

- **positivePoints**: Regions where **Control - Pathological** is significantly above SCC. These represent *areas where the Pathological group (AD) is hypoactive compared to the Control group*.
- **negativePoints**: Regions where **Control - Pathological** is significantly below SCC. These represent *areas where the Pathological group (AD) is hyperactive compared to the Control group*.

Make sure to check the order of Ya and Yb in the SCC computation before interpreting the results.

Usage

```
getPoints(sccResult)
```

Arguments

`sccResult`	A list containing SCC computation results from `scc.image`. The list should include at least: • `Z.band`: Matrix specifying grid positions. • `ind.inside.cover`: Indices of grid points inside the confidence band. • `scc`: 3D array containing computed SCC values.

Value

A named list:

- `positivePoints`: Data frame with coordinates where the **first group (Ya) had significantly lower activity than the second (Yb)**.
- `negativePoints`: Data frame with coordinates where the **first group (Ya) had significantly higher activity than the second (Yb)**.

See Also

scc.image for SCC computation.

Examples

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

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

# Show first extracted points (interpretation depends on SCC computation, see description)
head(significantPoints$positivePoints)  # Regions where Pathological is hypoactive vs. Control
head(significantPoints$negativePoints)  # Regions where Pathological is hyperactive vs. Control
```

---

getSPMbinary *Extract SPM-detected significant points from a binary NIfTI file*

Description

This function extracts voxel coordinates where pet = 1 (significant points) from a binary NIfTI file generated in an external SPM analysis. It filters data to retain only points in a given brain slice (z = paramZ).

The result is a data frame with detected points, structured identically to the output of getPoints. This allows direct comparison between SCC- and SPM-detected regions in calculateMetrics.

Usage

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

Arguments

niftiFile	character, the path to the binary NIfTI file.
paramZ	integer, the specific z-slice to extract. Default is 35.

Details

- This function processes externally generated SPM results into a compatible format for SCC analysis. - getDimensions can be used to verify the full coordinate grid.

Value

A data frame containing:

- x, y: Coordinates of detected points.

See Also

getPoints for SCC-based detection points. getDimensions for extracting voxel grid coordinates. 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 from PET image data to a functional data matrix format*

Description

This function transforms a database created by `databaseCreator` into a matrix format suitable for functional data analysis. Each row of the matrix represents a subject's PET data, formatted as a continuous line of data points.

Usage

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

Arguments

`database`	A data frame created by `databaseCreator` containing PET image data with columns for subject number, z, x, y, and pet values.
`paramZ`	The specific z-coordinate slice to analyze. Default is 35.
`useSequentialNumbering`	If `TRUE`, assigns sequential numbers instead of extracting from filenames.
`quiet`	If `TRUE`, suppresses progress messages.

Details

The function performs several operations:

1. Verifies that the specified z-slice exists in the database.
2. Automatically calculates matrix dimensions based on x and y coordinates.
3. Extracts PET intensity values for each subject at the specified z-slice.
4. Handles both control and pathological groups by recognizing subject identifiers.
5. Replaces any NaN values with zero to ensure matrix compatibility.

Typically follows `databaseCreator` and precedes `meanNormalization` in the neuroSCC pipeline.

Value

A matrix where each row represents the PET data from one subject, formatted as a continuous line of data points.

See Also

databaseCreator for creating the input database. meanNormalization for normalizing PET intensity values.

Examples

```
# Generate a database using databaseCreator
dataDir <- system.file("extdata", package = "neuroSCC")
controlPattern <- "^syntheticControl.*\\.nii.gz$"
databaseControls <- databaseCreator(pattern = controlPattern, control = TRUE, quiet = FALSE)

# Convert the database into a matrix format
matrixControls <- matrixCreator(databaseControls, paramZ = 35, quiet = FALSE)
dim(matrixControls)  # Show matrix dimensions
```

---

meanNormalization *Mean Average Normalization for Matrix Data*

Description

This function normalizes each row of the given matrix by its mean value. It divides each element in a row by the mean of that row, ignoring NA values. This normalization is a critical pre-processing step when comparing data from multiple sources to account for global intensity differences.

Usage

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

Arguments

matrixData	matrix, a matrix where each row represents data that needs to be normalized, typically the output from matrixCreator.
handleInvalidRows	character, specifies how to handle rows with invalid means (zero or NA). Options are: "warn" (default) warns and leaves row unnormalized, "error" stops with an error, or "omit" removes problematic rows from the result.
returnDetails	logical, if TRUE, returns a list containing the normalized matrix and additional diagnostic information. If FALSE (default), returns only the normalized matrix.
quiet	logical, if TRUE, suppresses console messages. Default is FALSE.

Details

The function iterates over each row of the input matrix, calculates the mean of the row, and then divides each element of the row by the calculated mean. This process adjusts for global intensity differences between rows in the matrix.

Value

A matrix where each row has been normalized by its mean value.

See Also

`matrixCreator` for creating the data matrix to be normalized.

Examples

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

# Normalize the matrix with detailed output
normalizationResult <- meanNormalization(matrixControls, returnDetails = TRUE, quiet = FALSE)
# Show problematic rows if any
if (length(normalizationResult$problemRows) == 0) {
  cat("No problematic rows detected.\n")
} else {
  print(normalizationResult$problemRows)
}
```

`neuroCleaner` *Clean and load data from NIFTI neuroimaging files*

Description

Loads a NIFTI-format neuroimaging file, converts it to a structured data frame, and organizes the data for further analysis. This function serves as the first step in the neuroimaging data processing pipeline, transforming raw PET data into a format suitable for functional data analysis. The neuroSCC package prepares and organizes neuroimaging data, while the actual Simultaneous Confidence Corridors (SCCs) are computed using functions from the ImageSCC package such as `ImageSCC::scc1g.image` or `ImageSCC::scc2g.image`.

Usage

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

Arguments

name	A character string specifying the path to the NIFTI file.
demo	An optional data frame containing demographic information for the participants. If provided, it should contain columns that match (case-insensitive): 'PPT', 'Group', 'Sex', and 'Age'. The 'PPT' column can contain participant IDs that match the name parameter for automatic matching, but if no match is found, the row specified by demoRow will be used. Default is NULL.
demoRow	An integer specifying which row of the demographic data to use when automatic matching fails or multiple matches are possible. Default is 1.

Details

The function performs several key operations:

1. Reads the NIFTI file using oro.nifti package functions.
2. Converts the 3D image data to a structured data frame.
3. Extracts the dimensional information (z, x, y coordinates).
4. Organizes the PET intensity values.
5. If demographic data is provided, merges it with the image data.

The resulting data frame serves as input for subsequent analysis functions in the neuroSCC pipeline, such as databaseCreator and matrixCreator, which prepare the data for eventual analysis with the ImageSCC package functions.

Value

A data frame with the following columns:

- If demographic data provided: PPT, Group, Sex, Age, z, x, y, pet
- If no demographic data: z, x, y, pet

Each row represents a voxel (3D pixel) from the image, with pet containing the intensity value at that location.

See Also

databaseCreator for creating databases from multiple NIFTI files.

readNIfTI for the underlying function used to read NIFTI files.

Examples

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

# Example 1: Without demographic data
petData <- neuroCleaner(niftiFile)
petData[sample(nrow(petData), 10), ] # Show 10 random voxels

# Example 2: With demographic data
demoFile <- system.file("extdata", "Demographics.csv", package = "neuroSCC")
demoData <- read.csv(demoFile, stringsAsFactors = FALSE, sep = ";")
petDataWithDemo <- neuroCleaner(niftiFile, demo = demoData)
petDataWithDemo[sample(nrow(petDataWithDemo), 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 extracted contour coordinates
if (length(contours) > 0) {
  head(contours[[1]])  # Show first few points of the main contour
}
```

`processROIs` *Process ROIs from a NIfTI file*

Description

This function processes Regions of Interest (ROIs) from a binary NIfTI file. It extracts voxel coordinates from the image and preserves the original structure, marking which voxels are part of the ROI.

This function is typically used in SCC evaluation, where detected SCC regions are compared with predefined ROIs in `calculateMetrics`.

Usage

```
processROIs(
  roiFile,
  region,
  number,
  save = TRUE,
  outputDir = "results/ROIs",
  verbose = TRUE
)
```

Arguments

`roiFile`	`character`, the path to the NIfTI file containing the ROI data.
`region`	`character`, the name of the ROI region (e.g., `"roi4"`).
`number`	`character`, the subject or group identifier (e.g., `"18"`).
`save`	`logical`, if `TRUE`, saves the processed ROIs as '.RDS' files. If `FALSE`, prints a preview in the console. Default is `TRUE`.
`outputDir`	`character`, directory where processed ROI tables will be saved.
`verbose`	`logical`, if `TRUE`, prints progress messages. Default is `TRUE`.

Details

The function reads the provided NIfTI file and extracts voxel data. It keeps all voxels, indicating whether each belongs to a ROI (`pet = 1`) or not (`pet = 0`).

Value

A data frame containing voxel-level ROI information, with columns:

- `group`: ROI identifier, composed of `region` + `number`.
- `z`, `x`, `y`: Voxel coordinates.
- `pet`: Binary indicator (`1` for ROI, `0` for non-ROI).

See Also

`calculateMetrics` for evaluating SCC detection performance. ROIs must be processed first to compare detected SCC voxels with predefined regions.

Examples

```
# Process an ROI NIfTI file (show results in console)
roiFile <- system.file("extdata", "ROIsample_Region2_18.nii.gz", package = "neuroSCC")
processedROI <- processROIs(roiFile, region = "Region2", number = "18", save = FALSE)
head(processedROI)  # Display first few rows
```

SCCcomp *Example SCC Computation Result*

Description

A precomputed example of a Simultaneous Confidence Corridor (SCC) analysis comparing a group of Pathological subjects against Controls, using the `scc.image` function.

This dataset serves as a compact and realistic representation of the kind of SCC object generated in neuroimaging comparisons, and is used internally in `getPoints` and `calculateMetrics` examples.

Usage

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

Format

A list of class `"image"` containing:

scc 3D array of SCC confidence bands, shape `[n, 2, alfa]`

Z.band Matrix of grid coordinates corresponding to evaluation points

ind.inside.cover Integer vector of indices identifying the SCC band region

V.est.a, V.est.b Vertex matrices for the domain triangulation (control and pathological)

Tr.est.a, Tr.est.b Triangle index matrices for the domain triangulation

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

d.est Spline degree used for mean estimation

r Smoothing parameter

Source

Simulated PET neuroimaging analysis for SCC evaluation.

See Also

getPoints, calculateMetrics, scc.image

Index