Package: scimagR (via r-universe)

November 3, 2026

Title Spinal Cord Injury Imaging Pipeline for MRI and CT Analysis in R

Version 0.1.0

Description An end-to-end pipeline for longitudinal MRI and CT analysis in spinal cord injury (SCI) research. Provides R wrappers for the Spinal Cord Toolbox (SCT) including SCIsegV2 lesion segmentation, DICOM anonymization and NIfTI conversion tools, SCI-specific clinical metrics (MSCC, compression ratio, tissue bridge quantification), implant artifact grading, and pipeline orchestration with built-in quality control. Integrates with workflowr for reproducible, publication-ready neuroimaging analyses.

License MIT + file LICENSE

URL https://github.com/cttir/scimagR

BugReports https://github.com/cttir/scimagR/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports cli (>= 3.0), dplyr (>= 1.1), ggplot2 (>= 3.4), glue (>= 1.6), readr, rlang (>= 1.0), tibble, tidyr, viridisLite, writexl

Suggests covr, here, knitr, patchwork, rmarkdown, rstudioapi, testthat (>= 3.0), withr, workflowr

SystemRequirements Spinal Cord Toolbox (>= 6.4) <https://spinalcordtoolbox.com>, dcm2niix <https://github.com/rordenlab/dcm2niix>, Python 3 with pydicom (pip install pydicom)

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website CTTIR/themakR

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev python3 libx11-dev zlib1g-dev

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

Date/Publication 2026-10-28 07:29:17 UTC

RemoteUrl https://github.com/CTTIR/scimagR

RemoteRef HEAD

RemoteSha 7cbb2555d64b81592b23a2eba787702721bc65d0

Contents

anonymize_dicom . . . 3
check_dcm2niix . . . 4
check_pydicom . . . 4
check_sct . . . 5
classify_artifact . . . 6
classify_phase . . . 6
compute_compression_ratio . . . 7
compute_csa_ratio . . . 8
compute_mscc . . . 8
convert_batch . . . 9
convert_dcm2niix . . . 10
coverage_matrix . . . 10
create_qc_log . . . 11
create_registry . . . 11
extract_dicom_metadata . . . 12
filter_evaluable . . . 13
integrity_summary . . . 14
log_exclusion . . . 14
pal_artifact . . . 15
pal_modality . . . 15
pal_phase . . . 16
plot_coverage . . . 16
plot_forest . . . 17
plot_spaghetti . . . 17
plot_violin_box . . . 18
save_fig . . . 19
save_rds . . . 20
save_table . . . 20
scale_artifact . . . 21
scale_modality . . . 21
scale_phase . . . 22
sci_create_project . . . 22
sci_pipeline_status . . . 23
sci_run_pipeline . . . 24
sct_analyze_lesion . . . 25
sct_generate_qc . . . 26
sct_label_vertebrae . . . 27
sct_process_segmentation . . . 28

sct_segment_lesion 29
sct_segment_sc 30
theme_sci 31
update_qc_log 31
validate_qc_log 32
validate_registry 32

Index **34**

`anonymize_dicom` *Anonymize DICOM Files*

Description

Replaces identifying header fields (PatientName, PatientID, PatientBirthDate, etc.) and removes private tags. Saves an ID mapping CSV to a user-specified secure location.

Usage

```
anonymize_dicom(
  dicom_dir,
  mapping_csv,
  prefix = "SCI",
  python = Sys.getenv("PYTHON_BIN", "python3")
)
```

Arguments

`dicom_dir`	Character. Root DICOM directory.
`mapping_csv`	Character. Path for ID mapping output. **Store securely!**
`prefix`	Character. Anonymized ID prefix. Default `"SCI"`.
`python`	Character. Python executable. Default `"python3"`.

Details

WARNING: This modifies DICOM files in-place. Create a backup first. The mapping CSV must NOT be stored in the project directory.

Value

Exit code (invisible).

Examples

```
## Not run:
anonymize_dicom("data/raw/dicom", "/secure/id_mapping.csv")

## End(Not run)
```

check_dcm2niix *Check dcm2niix Availability*

Description

Verifies that dcm2niix is installed and on PATH.

Usage

```
check_dcm2niix(verbose = TRUE)
```

Arguments

verbose	Logical. Print status message. Default TRUE.

Value

Named list with components: available, version, path.

Examples

```
## Not run:
check_dcm2niix()

## End(Not run)
```

check_pydicom *Check pydicom Availability*

Description

Verifies that Python 3 with pydicom is available.

Usage

```
check_pydicom(python = Sys.getenv("PYTHON_BIN", "python3"), verbose = TRUE)
```

Arguments

python	Character. Python executable. Default uses PYTHON_BIN environment variable or "python3".
verbose	Logical. Print status message. Default TRUE.

Value

Named list with components: available, version, path.

Examples

```
## Not run:
check_pydicom()

## End(Not run)
```

---

check_sct *Check Spinal Cord Toolbox Availability*

Description

Verifies that SCT is installed and on PATH. Returns version and path information.

Usage

```
check_sct(verbose = TRUE)
```

Arguments

verbose	Logical. Print status message. Default TRUE.

Value

Named list with components:

- available: Logical.
- version: Character. Version string or NA.
- path: Character. Path to sct_version executable or NA.

Examples

```
## Not run:
check_sct()

## End(Not run)
```

classify_artifact *Classify MRI Artifact Grade*

Description

Maps integer grades 0–4 to an ordered factor:

- 0: none
- 1: mild
- 2: moderate
- 3: severe
- 4: non-evaluable

Usage

```
classify_artifact(grade)
```

Arguments

grade	Integer vector (0–4). Values outside this range produce NA with a warning.

Value

Ordered factor with 5 levels: none < mild < moderate < severe < non-evaluable.

Examples

```
classify_artifact(c(0, 1, 2, 3, 4))
```

classify_phase *Classify SCI Phase Based on Days Post Trauma*

Description

Classification based on established SCI literature (Grassner et al. 2025, Freund et al. 2019):

- <= 1 day: hyperacute
- 2–14 days: acute
- 15–90 days: subacute
- 91–365 days: intermediate
- more than 365 days: chronic

Usage

```
classify_phase(days)
```

Arguments

days	Numeric vector. Days since trauma.

Value

Ordered factor with 5 levels: hyperacute < acute < subacute < intermediate < chronic.

Examples

```
classify_phase(c(0, 5, 30, 200, 400))
```

---

compute_compression_ratio

Compute Spinal Cord Compression Ratio

Description

Calculates the ratio of anterior-posterior to transverse diameter.

Usage

```
compute_compression_ratio(ap, transverse)
```

Arguments

ap	Numeric. Anterior-posterior diameter (mm).
transverse	Numeric. Transverse diameter (mm).

Value

Numeric. Compression ratio (unitless). NA if transverse <= 0 or any input is NA.

Examples

```
compute_compression_ratio(ap = 6.5, transverse = 12.0)
```

compute_csa_ratio *Compute Cross-Sectional Area Ratio*

Description

Calculates the ratio of CSA at the injury level to the reference CSA (average of adjacent levels).

Usage

```
compute_csa_ratio(csa_injury, csa_above, csa_below)
```

Arguments

csa_injury	Numeric. CSA at injury level (mm^2).
csa_above	Numeric. CSA one level above (mm^2).
csa_below	Numeric. CSA one level below (mm^2).

Value

Numeric. CSA ratio (unitless). NA if reference is <= 0 or any input is NA.

Examples

```
compute_csa_ratio(csa_injury = 45, csa_above = 70, csa_below = 72)
```

compute_mscc *Compute Maximum Spinal Cord Compression (MSCC)*

Description

Calculates MSCC using the formula:

$$MSCC = (1 - d_i/((d_a + d_b)/2)) \times 100$$

Usage

```
compute_mscc(di, da, db)
```

Arguments

di	Numeric. Sagittal AP diameter at injury level (mm).
da	Numeric. Sagittal AP diameter one level above (mm).
db	Numeric. Sagittal AP diameter one level below (mm).

Value

Numeric. MSCC percentage. NA if any input is NA or if the reference diameter is <= 0.

Examples

```
compute_mscc(di = 5.2, da = 8.1, db = 8.5)
compute_mscc(di = 0, da = 8, db = 8) # 100% compression
```

---

convert_batch *Batch Convert DICOM to NIfTI*

Description

Iterates over a registry and converts all sessions from DICOM to NIfTI.

Usage

```
convert_batch(
  registry,
  dicom_root,
  nifti_root,
  compress = TRUE,
  verbose = TRUE
)
```

Arguments

registry	Data frame with pat_id and session columns.
dicom_root	Character. Root DICOM directory.
nifti_root	Character. Root NIfTI output directory.
compress	Logical. Gzip output. Default TRUE.
verbose	Logical. Print progress. Default TRUE.

Value

Tibble with columns: pat_id, session, status, n_files.

Examples

```
## Not run:
convert_batch(registry, "data/raw/dicom", "data/nifti")

## End(Not run)
```

convert_dcm2niix *Convert DICOM to NIfTI*

Description

Converts a single session's DICOM files to NIfTI format using dcm2niix.

Usage

```
convert_dcm2niix(dicom_dir, output_dir, compress = TRUE, verbose = TRUE)
```

Arguments

dicom_dir	Character. Path to session DICOM directory.
output_dir	Character. Path for NIfTI output.
compress	Logical. Gzip output. Default TRUE.
verbose	Logical. Print progress. Default TRUE.

Value

Character vector of output NIfTI paths.

Examples

```
## Not run:
convert_dcm2niix("data/raw/dicom/SCI001/ses-01", "data/nifti/SCI001/ses-01")

## End(Not run)
```

coverage_matrix *Create a Modality Coverage Matrix*

Description

Shows which patients have which modalities at which timepoints.

Usage

```
coverage_matrix(registry)
```

Arguments

registry	Data frame. Validated registry with pat_id, session, modality, and optionally artifact_grade columns.

Value

Tibble in wide format with one row per patient and one column per session containing the modality.

Examples

```
## Not run:
reg <- validate_registry("imaging_registry.csv", strict = FALSE)
cm <- coverage_matrix(reg)

## End(Not run)
```

---

create_qc_log *Create a QC Segmentation Log*

Description

Creates an empty QC log tibble or reads an existing one from CSV.

Usage

```
create_qc_log(path = NULL)
```

Arguments

path	Character. Optional path to existing QC log CSV.

Value

Tibble with QC log columns: `pat_id`, `session`, `rater`, `date`, `seg_type`, `quality`, `notes`.

Examples

```
qc <- create_qc_log()
```

---

create_registry *Create an Imaging Registry from DICOM Metadata*

Description

Takes raw DICOM metadata (from `extract_dicom_metadata()`), restructures it into a proper imaging registry with computed fields (days_post_trauma, phase classification), and optionally saves to CSV.

Usage

```
create_registry(dicom_meta, trauma_dates, op_dates = NULL, output_csv = NULL)
```

Arguments

dicom_meta	Data frame or character path to DICOM metadata CSV.
trauma_dates	Named vector: names = pat_id, values = trauma date (Date or character coercible to Date).
op_dates	Named vector: names = pat_id, values = OP date. Optional.
output_csv	Character. Path to save registry. Optional.

Value

Tibble with registry columns.

Examples

```
## Not run:
meta <- read.csv("dicom_metadata.csv")
trauma <- c(SCI001 = "2024-01-15", SCI002 = "2024-02-20")
reg <- create_registry(meta, trauma)

## End(Not run)
```

---

extract_dicom_metadata

Extract DICOM Header Metadata

Description

Walks a DICOM directory tree and extracts key header fields (StudyDate, Modality, SeriesDescription, FieldStrength, etc.) from the first DICOM file per session.

Usage

```
extract_dicom_metadata(
  dicom_dir,
  output_csv,
  python = Sys.getenv("PYTHON_BIN", "python3")
)
```

Arguments

dicom_dir	Character. Root DICOM directory (expects <pat_id>/<session>/ structure).
output_csv	Character. Path for output CSV.
python	Character. Python executable. Default uses PYTHON_BIN environment variable or "python3".

Value

Character. Path to output CSV (invisible).

Examples

```
## Not run:
extract_dicom_metadata("data/raw/dicom", "data/metadata/dicom_metadata.csv")

## End(Not run)
```

`filter_evaluable` *Filter to Evaluable Imaging Sessions*

Description

Filters a dataset to sessions that meet artifact grade and modality criteria for analysis.

Usage

```
filter_evaluable(
  df,
  max_artifact = 2,
  modalities = c("MRT", "CT+MRT"),
  verbose = TRUE
)
```

Arguments

`df`	Data frame with `artifact_grade` and `modality` columns.
`max_artifact`	Integer. Maximum artifact grade to include. Default `2`.
`modalities`	Character vector. Modalities to include. Default `c("MRT", "CT+MRT")`.
`verbose`	Logical. Print exclusion summary. Default `TRUE`.

Value

Filtered data frame.

Examples

```
## Not run:
evaluable <- filter_evaluable(registry, max_artifact = 2)

## End(Not run)
```

integrity_summary *Print Data Integrity Summary*

Description

Reports dimensions, missing values per column, and duplicate count for a data frame. Returns the data frame invisibly for piping.

Usage

```
integrity_summary(df, id_cols = NULL)
```

Arguments

df	Data frame to summarize.
id_cols	Character vector. Columns to check for duplicates. Default NULL (skip duplicate check).

Value

Invisibly returns df.

Examples

```
## Not run:
registry |> integrity_summary(id_cols = c("pat_id", "session"))

## End(Not run)
```

log_exclusion *Log an Exclusion Step*

Description

Records before/after row counts and reason for an exclusion step. Useful for building a CONSORT-style exclusion flow.

Usage

```
log_exclusion(step, n_before, n_after, reason = "")
```

Arguments

step	Character. Name of the exclusion step.
n_before	Integer. Row count before filtering.
n_after	Integer. Row count after filtering.
reason	Character. Reason for exclusion.

Value

A 1-row tibble with columns: `step`, `n_before`, `n_after`, `n_excluded`, `reason`.

Examples

```
log_exclusion("Artifact filter", 100, 85, "artifact_grade > 2")
```

---

pal_artifact *Artifact Grade Colour Palette*

Description

Named character vector of 5 colours for MRI artifact grades: none (green), mild (yellow), moderate (orange), severe (red), non-evaluable (grey).

Usage

```
pal_artifact
```

Examples

```
pal_artifact
```

---

pal_modality *Modality Colour Palette*

Description

Named character vector of 3 colours for imaging modalities.

Usage

```
pal_modality
```

Examples

```
pal_modality
```

pal_phase *SCI Phase Colour Palette*

Description

Named character vector of 5 viridis-derived colours for SCI phases: hyperacute, acute, subacute, intermediate, chronic.

Usage

```
pal_phase
```

Examples

```
pal_phase
```

plot_coverage *Plot Modality Coverage Matrix*

Description

Creates a tile plot showing modality coverage per patient and session, with optional artifact grade overlay.

Usage

```
plot_coverage(registry)
```

Arguments

registry	Data frame. Validated registry.

Value

A ggplot2 object.

Examples

```
## Not run:
plot_coverage(registry)

## End(Not run)
```

plot_forest *Forest Plot*

Description

Creates a forest plot for comparing model estimates with confidence intervals. Useful for sensitivity analysis comparisons.

Usage

```
plot_forest(estimates_df, xlab = "Estimate", title = NULL, vline = 0)
```

Arguments

estimates_df	Data frame with columns: model, estimate, ci_lower, ci_upper.
xlab	Character. X-axis label. Default "Estimate".
title	Character. Plot title. Default NULL.
vline	Numeric. Position of reference line. Default 0.

Value

A ggplot2 object.

Examples

```
## Not run:
est <- data.frame(
  model = c("Full", "Sensitivity 1", "Sensitivity 2"),
  estimate = c(-2.1, -1.9, -2.3),
  ci_lower = c(-3.5, -3.4, -3.8),
  ci_upper = c(-0.7, -0.4, -0.8)
)
plot_forest(est)

## End(Not run)
```

plot_spaghetti *Spaghetti Plot*

Description

Shows individual patient trajectories over time with an optional group-level smoother.

Usage

```
plot_spaghetti(df, x, y, group, colour = NULL, smoother = TRUE, title = NULL)
```

Arguments

df	Data frame.
x	Character. Column name for x-axis (numeric, e.g., days).
y	Character. Column name for y-axis (numeric).
group	Character. Column name for grouping (e.g., "pat_id").
colour	Character. Column name for colour aesthetic. Default NULL.
smoother	Logical. Add group-level LOESS smoother. Default TRUE.
title	Character. Plot title. Default NULL.

Value

A ggplot2 object.

Examples

```
## Not run:
plot_spaghetti(df, x = "days_post_trauma", y = "csa",
               group = "pat_id", colour = "phase")

## End(Not run)
```

---

plot_violin_box *Violin + Box Plot*

Description

Creates a violin plot with embedded boxplot (no outlier points) and jittered individual data points. Uses viridis palette by default.

Usage

```
plot_violin_box(df, x, y, fill = x, title = NULL, ylab = y)
```

Arguments

df	Data frame.
x	Character. Column name for x-axis (categorical).
y	Character. Column name for y-axis (numeric).
fill	Character. Column name for fill aesthetic. Default same as x.
title	Character. Plot title. Default NULL.
ylab	Character. Y-axis label. Default same as y.

Value

A ggplot2 object.

Examples

```
## Not run:
plot_violin_box(df, x = "phase", y = "mscc", fill = "phase")

## End(Not run)
```

---

save_fig *Save Figure with Standard Conventions*

Description

Saves a ggplot2 plot at publication-ready resolution with white background.

Usage

```
save_fig(plot, filename, path, width = 7, height = 5, dpi = 600)
```

Arguments

plot	A ggplot2 object.
filename	Character. Output filename (e.g., "fig1_mscc.png").
path	Character. Output directory.
width	Numeric. Figure width in inches. Default 7.
height	Numeric. Figure height in inches. Default 5.
dpi	Integer. Resolution. Default 600.

Value

Character. Full path to saved file (invisible).

Examples

```
## Not run:
p <- ggplot2::ggplot(mtcars, ggplot2::aes(wt, mpg)) + ggplot2::geom_point()
save_fig(p, "test_plot.png", tempdir())

## End(Not run)
```

save_rds *Save R Object as RDS*

Description

Saves any R object as .rds with a confirmation message.

Usage

```
save_rds(obj, filename, path)
```

Arguments

obj	Any R object.
filename	Character. Output filename (e.g., "model_fit.rds").
path	Character. Output directory.

Value

Character. Full path to saved file (invisible).

Examples

```
## Not run:
save_rds(mtcars, "cars.rds", tempdir())

## End(Not run)
```

save_table *Save Table as Excel*

Description

Saves a data frame as .xlsx via writexl.

Usage

```
save_table(df, filename, path)
```

Arguments

df	A data frame.
filename	Character. Output filename (e.g., "table1.xlsx").
path	Character. Output directory.

Value

Character. Full path to saved file (invisible).

Examples

```
## Not run:
save_table(mtcars, "cars.xlsx", tempdir())

## End(Not run)
```

---

scale_artifact *Colour and Fill Scales for Artifact Grades*

Description

Colour and Fill Scales for Artifact Grades

Usage

```
scale_colour_artifact(...)

scale_fill_artifact(...)
```

Arguments

...	Additional arguments passed to ggplot2::scale_colour_manual() / ggplot2::scale_fill_manual(

Value

A ggplot2 scale object.

---

scale_modality *Colour and Fill Scales for Imaging Modalities*

Description

Colour and Fill Scales for Imaging Modalities

Usage

```
scale_colour_modality(...)

scale_fill_modality(...)
```

Arguments

...	Additional arguments passed to ggplot2::scale_colour_manual() / ggplot2::scale_fill_manual(

Value

A ggplot2 scale object.

`scale_phase` *Colour and Fill Scales for SCI Phases*

Description

Convenience wrappers around `ggplot2::scale_colour_manual()` and `ggplot2::scale_fill_manual()` using pal_phase.

Usage

```
scale_colour_phase(...)

scale_fill_phase(...)
```

Arguments

`...`	Additional arguments passed to `ggplot2::scale_colour_manual()` / `ggplot2::scale_fill_manual(`

Value

A ggplot2 scale object.

Examples

```
## Not run:
ggplot(df, aes(x, y, colour = phase)) + geom_point() + scale_colour_phase()

## End(Not run)
```

`sci_create_project` *Create a New SCI MRI/CT Analysis Project*

Description

Sets up a complete workflowr project with full directory structure, pre-configured YAML files, template Rmd files, CSV templates, and a `.gitignore` tuned for neuroimaging projects.

Usage

```
sci_create_project(
  path,
  title = "SCI Imaging Analysis",
  author = "Author Name",
  seed = 20260411L,
  open = TRUE
)
```

Arguments

path	Character. Directory for the new project.
title	Character. Project title.
author	Character. Author name.
seed	Integer. Reproducibility seed. Default 20260411.
open	Logical. Open project in RStudio. Default TRUE.

Value

Character. Path to created project (invisible).

Examples

```
## Not run:
sci_create_project("~/projects/my-sci-study",
  title = "Cervical SCI Longitudinal MRI Analysis",
  author = "Raban Heller")

## End(Not run)
```

---

sci_pipeline_status *Get Pipeline Status*

Description

Scans the project directory and reports which processing steps are complete for each patient and session.

Usage

```
sci_pipeline_status(project_dir)
```

Arguments

project_dir	Character. Path to workflowr project root.

Value

Tibble with columns: `pat_id`, `session`, `converted`, `sc_seg`, `lesion_seg`, `vert_labels`, `qc_done`, `params_extracted`.

Examples

```
## Not run:
sci_pipeline_status("~/projects/my-sci-study")

## End(Not run)
```

`sci_run_pipeline` *Run the SCI Analysis Pipeline*

Description

Orchestrates all external tool steps in sequence:

1. DICOM metadata extraction
2. Anonymization (if requested)
3. DICOM to NIfTI conversion
4. Spinal cord segmentation
5. Lesion segmentation (SCIsegV2)
6. Vertebral labeling
7. Parameter extraction
8. Merge & export

Usage

```
sci_run_pipeline(
  project_dir,
  steps = 1:8,
  anonymize = FALSE,
  overwrite = FALSE,
  verbose = TRUE
)
```

Arguments

`project_dir`	Character. Path to workflowr project root.
`steps`	Integer vector. Which steps to run. Default `1:8`.
`anonymize`	Logical. Run anonymization step. Default `FALSE`.
`overwrite`	Logical. Re-process existing outputs. Default `FALSE`.
`verbose`	Logical. Print progress. Default `TRUE`.

Details

Each step checks for existing outputs (idempotent). The pipeline can be interrupted and resumed at any step.

Value

Tibble with pipeline status per step.

Examples

```
## Not run:
sci_run_pipeline("~/projects/my-sci-study", steps = 1:3)

## End(Not run)
```

---

sct_analyze_lesion *Extract Lesion Parameters*

Description

Runs `sct_analyze_lesion` to compute lesion volume, length, max axial damage ratio, and tissue bridge measurements.

Usage

```
sct_analyze_lesion(lesion_seg, sc_seg, output_dir = NULL, verbose = TRUE)
```

Arguments

`lesion_seg`	Character. Path to lesion segmentation NIfTI.
`sc_seg`	Character. Path to spinal cord segmentation NIfTI.
`output_dir`	Character. Directory for CSV output.
`verbose`	Logical. Print progress. Default `TRUE`.

Value

Character. Path to output CSV, or `NA` on failure.

Examples

```
## Not run:
sct_analyze_lesion("sub-01_lesion_seg.nii.gz", "sub-01_seg.nii.gz")

## End(Not run)
```

`sct_generate_qc` *Generate SCT Quality Control Report*

Description

Creates an HTML QC report overlaying a segmentation on the original image.

Usage

```
sct_generate_qc(
  input,
  segmentation,
  qc_dir,
  process = "sct_deepseg",
  verbose = TRUE
)
```

Arguments

`input`	Character. Path to original NIfTI image.
`segmentation`	Character. Path to segmentation overlay.
`qc_dir`	Character. Directory for QC HTML output.
`process`	Character. SCT process name for labeling. Default `"sct_deepseg"`.
`verbose`	Logical. Print progress. Default `TRUE`.

Value

Exit code (invisible).

Examples

```
## Not run:
sct_generate_qc("sub-01_T2w.nii.gz", "sub-01_T2w_seg.nii.gz", "qc/")

## End(Not run)
```

`sct_label_vertebrae` *Label Vertebral Levels*

Description

Runs SCT's vertebral labeling on a spinal cord segmentation.

Usage

```
sct_label_vertebrae(
  input,
  segmentation,
  contrast = "t2",
  output_dir = NULL,
  overwrite = FALSE,
  verbose = TRUE
)
```

Arguments

`input`	Character. Path to T2w NIfTI file.
`segmentation`	Character. Path to spinal cord segmentation.
`contrast`	Character. MRI contrast type. Default `"t2"`.
`output_dir`	Character. Directory for output labels. Defaults to the directory of `input`.
`overwrite`	Logical. Re-run if output exists? Default `FALSE`.
`verbose`	Logical. Print progress. Default `TRUE`.

Value

Character. Path to vertebral label NIfTI, or `NA` on failure.

Examples

```
## Not run:
sct_label_vertebrae("sub-01_T2w.nii.gz", "sub-01_T2w_seg.nii.gz")

## End(Not run)
```

`sct_process_segmentation` *Extract Morphometric Parameters*

Description

Runs `sct_process_segmentation` to compute cross-sectional area (CSA), AP diameter, and transverse diameter per vertebral level.

Usage

```
sct_process_segmentation(
  sc_seg,
  vert_labels,
  output_dir = NULL,
  perslice = FALSE,
  verbose = TRUE
)
```

Arguments

`sc_seg`	Character. Path to spinal cord segmentation NIfTI.
`vert_labels`	Character. Path to vertebral labels NIfTI.
`output_dir`	Character. Directory for CSV output.
`perslice`	Logical. Compute per-slice metrics. Default `FALSE`.
`verbose`	Logical. Print progress. Default `TRUE`.

Value

Character. Path to output CSV, or `NA` on failure.

Examples

```
## Not run:
sct_process_segmentation("sub-01_seg.nii.gz", "sub-01_labels-disc.nii.gz")

## End(Not run)
```

`sct_segment_lesion` *Segment SCI Lesion*

Description

Runs SCIsegV2 lesion segmentation model via SCT.

Usage

```
sct_segment_lesion(
  input,
  output = NULL,
  output_dir = NULL,
  overwrite = FALSE,
  verbose = TRUE,
  qc_dir = NULL
)
```

Arguments

`input`	Character. Path to T2w NIfTI file (.nii.gz).
`output`	Character. Path for output segmentation. Auto-generated if `NULL`.
`output_dir`	Character. Directory for output. Alternative to `output`.
`overwrite`	Logical. Re-run if output exists? Default `FALSE`.
`verbose`	Logical. Print progress. Default `TRUE`.
`qc_dir`	Character. If provided, auto-generate QC report.

Value

Character. Path to output lesion segmentation, or `NA` on failure.

Examples

```
## Not run:
sct_segment_lesion("sub-01_T2w.nii.gz")

## End(Not run)
```

`sct_segment_sc` *Segment Spinal Cord*

Description

Runs SCT's contrast-agnostic spinal cord segmentation model.

Usage

```
sct_segment_sc(
  input,
  output = NULL,
  output_dir = NULL,
  overwrite = FALSE,
  verbose = TRUE,
  qc_dir = NULL
)
```

Arguments

`input`	Character. Path to NIfTI file (.nii.gz).
`output`	Character. Path for output segmentation. Auto-generated if `NULL`.
`output_dir`	Character. Directory for output. Alternative to `output`.
`overwrite`	Logical. Re-run if output exists? Default `FALSE` (idempotent).
`verbose`	Logical. Print progress. Default `TRUE`.
`qc_dir`	Character. If provided, auto-generate QC report.

Value

Character. Path to output segmentation file, or `NA` on failure.

Examples

```
## Not run:
sct_segment_sc("sub-01_T2w.nii.gz")

## End(Not run)
```

theme_sci *SCI Publication Theme*

Description

A clean ggplot2 theme based on `ggplot2::theme_bw()`, optimized for publication-ready neuroimaging figures. Uses Arial font, no minor gridlines, and a bottom legend.

Usage

```
theme_sci(base_size = 11, base_family = "Arial")
```

Arguments

`base_size`	Numeric. Base font size. Default `11`.
`base_family`	Character. Base font family. Default `"Arial"`.

Value

A ggplot2 theme object.

Examples

```
## Not run:
library(ggplot2)
ggplot(mtcars, aes(wt, mpg)) + geom_point() + theme_sci()

## End(Not run)
```

update_qc_log *Update a QC Log*

Description

Adds one or more QC entries to a QC log.

Usage

```
update_qc_log(qc_log, entries, output_csv = NULL)
```

Arguments

`qc_log`	Tibble. Existing QC log (from `create_qc_log()`).
`entries`	Data frame. New entries with the same columns as `qc_log`.
`output_csv`	Character. Optional path to save updated log.

Value

Updated QC log tibble.

Examples

```
## Not run:
qc <- create_qc_log()
entry <- tibble::tibble(
  pat_id = "SCI001", session = "ses-01", rater = "RH",
  date = Sys.Date(), seg_type = "sc", quality = "good", notes = ""
)
qc <- update_qc_log(qc, entry)

## End(Not run)
```

---

validate_qc_log *Validate a QC Log*

Description

Checks required columns and valid quality values.

Usage

```
validate_qc_log(qc_log)
```

Arguments

qc_log	Data frame or path to CSV.

Value

Invisibly returns the validated QC log.

---

validate_registry *Validate an Imaging Registry*

Description

Checks: required columns present, no duplicate pat_id x session, dates are valid, days_post_trauma >= 0, artifact_grade in 0–4, modality in allowed values.

Usage

```
validate_registry(registry, strict = TRUE)
```

Arguments

registry	Data frame or character path to CSV.
strict	Logical. Stop on errors (TRUE) or warn (FALSE). Default TRUE.

Value

Invisibly returns the validated registry tibble.

Examples

```
## Not run:
validate_registry("data/metadata/imaging_registry.csv")

## End(Not run)
```

Index

anonymize_dicom, 3

check_dcm2niix, 4
check_pydicom, 4
check_sct, 5
classify_artifact, 6
classify_phase, 6
compute_compression_ratio, 7
compute_csa_ratio, 8
compute_mscc, 8
convert_batch, 9
convert_dcm2niix, 10
coverage_matrix, 10
create_qc_log, 11
create_qc_log(), *31*
create_registry, 11

extract_dicom_metadata, 12
extract_dicom_metadata(), *11*

filter_evaluable, 13

ggplot2::scale_colour_manual(), *21, 22*
ggplot2::scale_fill_manual(), *21, 22*
ggplot2::theme_bw(), *31*

integrity_summary, 14

log_exclusion, 14

pal_artifact, 15
pal_modality, 15
pal_phase, 16, *22*
plot_coverage, 16
plot_forest, 17
plot_spaghetti, 17
plot_violin_box, 18

save_fig, 19
save_rds, 20
save_table, 20

scale_artifact, 21
scale_colour_artifact *(*scale_artifact*)*, 21
scale_colour_modality *(*scale_modality*)*, 21
scale_colour_phase *(*scale_phase*)*, 22
scale_fill_artifact *(*scale_artifact*)*, 21
scale_fill_modality *(*scale_modality*)*, 21
scale_fill_phase *(*scale_phase*)*, 22
scale_modality, 21
scale_phase, 22
sci_create_project, 22
sci_pipeline_status, 23
sci_run_pipeline, 24
sct_analyze_lesion, 25
sct_generate_qc, 26
sct_label_vertebrae, 27
sct_process_segmentation, 28
sct_segment_lesion, 29
sct_segment_sc, 30

theme_sci, 31

update_qc_log, 31

validate_qc_log, 32
validate_registry, 32