Package 'BoneDensityMapping'

July 25, 2025

Type Package

Title Maps Bone Densities from CT Scans to Surface Models

Version 0.1.1

Maintainer Scott Telfer <scott.telfer@gmail.com>

Description Allows local bone density estimates to be derived from CT data and mapped to 3D bone models in a reproducible manner. Processing can be performed at the individual bone or group level. Also includes tools for visualizing the bone density estimates. Example methods are described in Telfer et al., (2021) <doi:10.1002/jor.24792>, Telfer et al., (2021) <doi:10.1016/j.jse.2021.05.011>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports cowplot, ggplot2, ggpubr, methods, oro.nifti, ptinpoly, rdist, rjson, concaveman, geometry, sp, jsonlite, rgl, RNifti, Rvcg, FNN, nat

Depends R (>= 2.10)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Scott Telfer [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-0104-4027>), Lucas Lacambra [aut]

Repository CRAN

Date/Publication 2025-07-25 10:10:14 UTC

Contents

`bone_scan_check` *Check if surface model is fully contained within scan volume*

Description

Check if surface model is fully contained within scan volume

Usage

```
bone_scan_check(surface_mesh, nifti, return_limits = FALSE)
```

Arguments

`surface_mesh`	mesh object (class `mesh3d`) or numeric matrix/dataframe of vertex coordinates (cols: X, Y, Z)
`nifti`	NIfTI image object representing CT scan.
`return_limits`	Logical. If TRUE returns a summary of the bounding boxes of the scan and mesh

Value

If any vertices lie outside the scan volume, it raises an error.

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url2, bone_filepath, mode = "wb")
 surface_mesh <- import_mesh(bone_filepath)
 bone_scan_check(surface_mesh, nifti, return_limits = TRUE)
```

---

color_bar *Produce stand alone color bar*

Description

Produce stand alone color bar

Usage

```
color_bar(
  colors,
  mini,
  maxi,
  orientation = "vertical",
  breaks,
  title = "",
  text_size = 11,
  plot = TRUE
)
```

Arguments

colors	String
mini	Numeric
maxi	Numeric
orientation	"horizontal" or "vertical"
breaks	Numeric vector
title	String
text_size	Numeric
plot	Logical

Value

ggplot object

Examples

```
colors <- c("darkblue", "blue", "lightblue", "green", "yellow", "red", "pink")
color_bar(colors, 0, 2000, breaks = c(0, 500, 1000, 1500, 2000))
```

`color_mapping` *maps numeric values to a color*

Description

maps numeric values to a color

Usage

```
color_mapping(x, maxi, mini, color_sel)
```

Arguments

`x`	Vector.
`maxi`	Numeric. Maximum value. Defaults to maximum value in vector. Can be useful to set this manually if there are outliers in the scan data
`mini`	Numeric. Minimum value. Defaults to minimum value in vector. Can be useful to set this manually if there are outliers in the scan data
`color_sel`	Vector. Colors to use for map. Defaults to a scale of "grey", "blue", "green", "yellow", "orange", "red", "pink".

Value

Vector of hex color values same length as x

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url2, bone_filepath, mode = "wb")
 surface_mesh <- import_mesh(bone_filepath)
 mat_peak <- voxel_point_intersect(surface_mesh, nifti, ct_eqn = "linear",
                                   ct_params = c(68.4, 1.106),)
 colors <- color_mapping(mat_peak)
```

color_mesh *Takes a density vector mapped to standardized coordinates and maps it to a surface mesh for visualization.*

Description

Takes a density vector mapped to standardized coordinates and maps it to a surface mesh for visualization.

Usage

```
color_mesh(
  surface_mesh,
  template_pts,
  density_vector,
  maxi = NULL,
  mini = NULL,
  export_path,
  color_sel
)
```

Arguments

surface_mesh	Mesh object
template_pts	Matrix
density_vector	Vector
maxi	Numeric
mini	Numeric
export_path	Character
color_sel	String

Value

mesh3d object with added color dimension

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.1/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
```

```
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
bone_filepath <- tempfile(fileext = ".stl")
download.file(url2, bone_filepath, mode = "wb")
surface_mesh <- import_mesh(bone_filepath)
landmark_path <- system.file("extdata", "test_femur.mrk.json",
                             package = "BoneDensityMapping")
landmarks <- import_lmks(landmark_path)
mapped_coords <- surface_points_template(surface_mesh, landmarks,
                                         no_surface_sliders = 1000)
dens <- voxel_point_intersect(mapped_coords, nifti,
                              ct_eqn = "linear",
                              ct_params = c(68.4, 1.106))
colored_mesh <- color_mesh(surface_mesh, mapped_coords, dens)
```

`ct_calibration` *Sigma beta CT calculations*

Description

Sigma beta CT calculations

Usage

```
ct_calibration(ct_nos, calibration_type, params)
```

Arguments

`ct_nos`	Numeric vector. CT numbers from scan
`calibration_type`	String. Currently only linear supported.
`params`	Numeric vector. beta and sigma values for calibration eqn

Value

Vector with estimated density values in mg/cm^3

Author(s)

Scott Telfer <scott.telfer@gmail.com>

fill_bone_points *Fills bone with orthogonally spaced points for internal analysis*

Description

Fills bone with orthogonally spaced points for internal analysis

Usage

```
fill_bone_points(surface_mesh, spacing)
```

Arguments

surface_mesh	Mesh object
spacing	Numeric

Value

Matrix with internal point coordinates

Examples

```
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
bone_filepath <- tempfile(fileext = ".stl")
download.file(url, bone_filepath, mode = "wb")
surface_mesh <- import_mesh(bone_filepath)
internal_fill <- fill_bone_points(surface_mesh, 2)
```

import_lmks *import landmark coordinates*

Description

import landmark coordinates

Usage

```
import_lmks(landmark_path)
```

Arguments

landmark_path	String. File path to landmark data. .json or .fcsv format

Value

dataframe. Columns are landmark name, x, y, and z coordinates

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
landmark_path <- system.file("extdata", "test_femur.mrk.json",
                             package = "BoneDensityMapping")
landmarks <- import_lmks(landmark_path)
```

---

import_mesh *import surface mesh*

Description

import surface mesh

Usage

```
import_mesh(surface_mesh_filepath)
```

Arguments

surface_mesh_filepath	String. File path to bone models. .stl or .ply

Value

mesh object

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download bone model
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url, bone_filepath, mode = "wb")
 import_mesh(bone_filepath)
```

import_scan *import CT scan*

Description

import CT scan

Usage

```
import_scan(scan_filepath)
```

Arguments

scan_filepath	String. File path to CT scan data. Should be .nii or .nrrd

Value

scan object

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 import_scan(scan_filepath)
```

landmark_check *Check landmarks are close to the mesh*

Description

Check landmarks are close to the mesh

Usage

```
landmark_check(surface_mesh, landmarks, threshold = 1)
```

Arguments

surface_mesh	mesh object
landmarks	Dataframe. Columns are landmark name, x, y, and z coords
threshold	Numeric. Distance landmark can be from surface without warning being thrown

Value

String. Returns a message warning that landmarks are not on bone surface

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download bone model
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url, bone_filepath, mode = "wb")
 surface_mesh <- import_mesh(bone_filepath)
 landmark_path <- system.file("extdata", "test_femur.fcsv",
                              package = "BoneDensityMapping")
 landmarks <- import_lmks(landmark_path)
 landmark_check(surface_mesh, landmarks, threshold = 1.0)
```

---

plot_cross_section_bone

Plot Cross-Sectional Bone Visualization in 3D

Description

Visualizes a 3D cross-section of a bone using surface mesh and internal density (fill) points. Clips the surface mesh at a given axis and value, and overlays a 2D projection of internal density.

Usage

```
plot_cross_section_bone(
  surface_mesh,
  surface_colors = NULL,
  fill_coords,
  fill_colors,
  slice_axis,
  slice_val,
  slice_thickness = 1,
  IncludeSurface = FALSE,
  title = "Bone Cross-Section",
  userMat = NULL,
  legend = TRUE,
  legend_color_sel = NULL,
  legend_maxi = NULL,
  legend_mini = NULL
)
```

Arguments

surface_mesh	A ‘mesh3d‘ object representing the outer surface of the bone.
surface_colors	Optional. A vector of colors for each vertex of the surface mesh. If NULL, uses mesh’s own material colors.
fill_coords	A numeric matrix of internal fill point coordinates.
fill_colors	A vector of colors corresponding to fill points.
slice_axis	Character. ‘’x’‘, ‘’y’‘, or ‘’z’‘. Axis along which to slice.
slice_val	Numeric (0 to 1). Relative slice location along selected axis.
slice_thickness	Numeric. Width of the slice (default = 1).
IncludeSurface	Logical. Whether to include the clipped surface mesh.
title	Character. Title for the plot.
userMat	Optional. A 4x4 matrix controlling view orientation.
legend	Logical. Optional color bar.
legend_color_sel	Optional character with color gradient
legend_maxi	Numeric. Maximum bone density.
legend_mini	Numeric. Minimum bone density.

Value

Generates an ‘rgl‘ plot

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.1/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url2, bone_filepath, mode = "wb")
 surface_mesh <- import_mesh(bone_filepath)
 landmark_path <- system.file("extdata", "test_femur.mrk.json",
                              package = "BoneDensityMapping")
 landmarks <- import_lmks(landmark_path)
 mapped_coords <- surface_points_template(surface_mesh, landmarks,
                                          no_surface_sliders = 100)
 mat_peak <- voxel_point_intersect(mapped_coords, nifti)
 colored_mesh <- color_mesh(surface_mesh, mapped_coords, mat_peak)
 internal_fill <- fill_bone_points(surface_mesh, 3)
 internal_density <- voxel_point_intersect(internal_fill, nifti,
                                           ct_eqn = "linear",
                                           ct_params = c(68.4, 1.106))
 internal_colors <- color_mapping(internal_density)
```

```
plot_cross_section_bone(colored_mesh, surface_colors = NULL,
                        internal_fill, internal_colors, slice_axis = 'x',
                        slice_val = 0.5)
```

`plot_mesh` *plot mesh*

Description

plot mesh

Usage

```
plot_mesh(
  surface_mesh,
  density_color = NULL,
  title = NULL,
  legend = TRUE,
  legend_color_sel = NULL,
  legend_maxi = 2000,
  legend_mini = 0,
  userMat = NULL
)
```

Arguments

`surface_mesh`	Mesh object
`density_color`	Vector. Colors mapped from density values.
`title`	String. Plot title.
`legend`	Logical. Optional color bar.
`legend_color_sel`	Optional character with color gradient
`legend_maxi`	Numeric. Maximum bone density.
`legend_mini`	Numeric. Minimum bone density.
`userMat`	Optional matrix. Controls graph orientation.

Value

plot of mesh with color

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.1/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url2, bone_filepath, mode = "wb")
 surface_mesh <- import_mesh(bone_filepath)
 vertices <- t(surface_mesh$vb)[, c(1:3)]
 landmark_path <- system.file("extdata", "test_femur.mrk.json",
                              package = "BoneDensityMapping")
 landmarks <- import_lmks(landmark_path)
 mapped_coords <- surface_points_template(surface_mesh, landmarks,
                                          no_surface_sliders = 5000)
 mat_peak <- surface_normal_intersect(surface_mesh, mapped_coords,
                                      normal_dist = 3.0, nifti, rev_y=FALSE)
 color_mesh <- color_mesh(surface_mesh, mapped_coords, mat_peak, maxi = 2000,
                          mini = 0)
 plot <- plot_mesh(color_mesh)
```

`reorientate_landmarks` *Reorientate landmarks*

Description

Reorientate landmarks

Usage

```
reorientate_landmarks(landmark_path, x = 1, y = 1, z = 1)
```

Arguments

`landmark_path`	String
`x`	Integer Value to apply to convert mesh i.e. -1 will mirror x coords
`y`	Integer Value to apply to convert mesh i.e. -1 will mirror y coords
`z`	Integer Value to apply to convert mesh i.e. -1 will mirror z coords

Value

Overwritten landmark file

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
landmark_path <- system.file("extdata", "test_femur.mrk.json",
                             package = "BoneDensityMapping")
reoriented_landmarks <- reorientate_landmarks(landmark_path)
```

`rm_local_sig` *local significance*

Description

local significance

Usage

```
rm_local_sig(vertices, sig_vals, changes, sig_level = 0.05, dist)
```

Arguments

`vertices`	Matrix
`sig_vals`	Numeric vector
`changes`	Numeric vector
`sig_level`	Numeric. Default 0.05
`dist`	Numeric. Distance to check for vertices

Value

Numeric vector

Author(s)

Scott Telfer <scott.telfer@gmail.com>

`surface_normal_intersect`
Find material properties of bone at surface point using surface normal

---

Description

Find material properties of bone at surface point using surface normal

Usage

```
surface_normal_intersect(
  surface_mesh,
  mapped_coords = NULL,
  normal_dist = 3,
  nifti,
  ct_eqn = NULL,
  ct_params = NULL,
  rev_x = FALSE,
  rev_y = FALSE,
  rev_z = FALSE,
  check_in_vol = FALSE
)
```

Arguments

surface_mesh	Mesh object
mapped_coords	Data frame. 3D coords of remapped surface points. If NULL, surface_mesh vertices will be used
normal_dist	Numeric. Distance surface normal should penetrate surface
nifti	Nifti CT scan image
ct_eqn	String. Equation to use for density calibration. Currently "linear" supported.
ct_params	Numeric vector. Calibration parameters for density calculation. For linear, first value is beta coefficient (y intercept), second value is sigma coefficient (gradient)
rev_x	Logical. Reverses x voxel coordinates
rev_y	Logical. Reverses y voxel coordinates
rev_z	Logical. Reverses z voxel coordinates
check_in_vol	Logical. Include check that model is in scans volume and print dimensions

Value

Vector. Vector with value for each point on surface

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
```

```
download.file(url2, bone_filepath, mode = "wb")
surface_mesh <- import_mesh(bone_filepath)
landmark_path <- system.file("extdata", "test_femur.mrk.json",
                             package = "BoneDensityMapping")
landmarks <- import_lmks(landmark_path)
mapped_coords <- surface_points_template(surface_mesh, landmarks,
                                         no_surface_sliders = 1000)
mat_peak <- surface_normal_intersect(surface_mesh, normal_dist = 3.0,
                                     nifti = nifti, ct_eqn = "linear",
                                     ct_params = c(68.4, 1.106))
```

`surface_points_new` *New mapped surface points from template*

Description

New mapped surface points from template

Usage

```
surface_points_new(
  surface_mesh,
  landmarks,
  template,
  mirror = FALSE,
  plot_check = FALSE
)
```

Arguments

`surface_mesh`	List. Mesh data imported via ply_import function
`landmarks`	Data frame. Contains 3D coords of landmarks
`template`	Data frame. 3D coords of remapped surface points
`mirror`	Logical or character. Set to "x", "y", or "z" to mirror the mesh and landmarks across that axis before remapping.
`plot_check`	Logical. If TRUE, generates a 3D plot showing the mirrored mesh, mirrored landmarks, remapped surface points, and original template points to visually verify correct orientation and laterality.

Value

Data frame. 3D coords of remapped surface points

Author(s)

Scott Telfer <scott.telfer@gmail.com> Adapted from geomorph

Examples

```
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/SCAP001.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url, bone_filepath, mode = "wb")
 scap_001_mesh <- import_mesh(bone_filepath)
 landmark_path <- system.file("extdata", "SCAP001_landmarks.fcsv",
                              package = "BoneDensityMapping")
 scap_001_lmk <- import_lmks(landmark_path)
 template_coords <- surface_points_template(scap_001_mesh, scap_001_lmk,
                                            1000)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/SCAP002.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url2, bone_filepath, mode = "wb")
 scap_002_mesh <- import_mesh(bone_filepath)
 landmark_path <- system.file("extdata", "SCAP002_landmarks.fcsv",
                              package = "BoneDensityMapping")
 scap_002_lmk <- import_lmks(landmark_path)
 scap_002_remapped <- surface_points_new(scap_002_mesh, scap_002_lmk,
                                         template_coords, mirror = "x",
                                         plot_check = FALSE)
```

---

surface_points_template

Redefine surface points. Adds additional surface points ("sliders") that are spatially distributed across the mesh surface. Adapted from geomorph

---

Description

Redefine surface points. Adds additional surface points ("sliders") that are spatially distributed across the mesh surface. Adapted from geomorph

Usage

```
surface_points_template(surface_mesh, landmarks, no_surface_sliders)
```

Arguments

surface_mesh	Mesh object
landmarks	Data frame with landmark coordinates (columns: ID, x, y, z)
no_surface_sliders	Numeric. No. of surface points to generate

Value

Data frame. 3D coordinates for the combined set of original landmarks and the new surface points

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
bone_filepath <- tempfile(fileext = ".stl")
download.file(url, bone_filepath, mode = "wb")
surface_mesh <- import_mesh(bone_filepath)
landmark_path <- system.file("extdata", "test_femur.mrk.json",
                             package = "BoneDensityMapping")
landmarks <- import_lmks(landmark_path)
mapped_coords <- surface_points_template(surface_mesh, landmarks, 1000)
```

---

voxel_point_intersect *Finds material properties of bone at any point*

Description

Finds material properties of bone at any point

Usage

```
voxel_point_intersect(
  vertex_coords,
  nifti,
  ct_eqn = NULL,
  ct_params = NULL,
  rev_x = FALSE,
  rev_y = FALSE,
  rev_z = FALSE,
  check_in_vol = FALSE
)
```

Arguments

vertex_coords	Matrix
nifti	nifti object
ct_eqn	String. Equation to use for density calibration. Currently only "linear" is supported.
ct_params	Numeric vector. Calibration parameters for density calculation. For linear, first value is beta coefficient (y intercept), second value is sigma coefficient (gradient)
rev_x	Logical. Reverses x voxel coordinates
rev_y	Logical. Reverses y voxel coordinates
rev_z	Logical. Reverses z voxel coordinates
check_in_vol	Logical. Include check that model is in scans volume and print dimensions

Value

Vector. Vector with value for each point on surface

Author(s)

Scott Telfer <scott.telfer@gmail.com>

Examples

```
 # Download CT scan
url <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_hip.nii.gz"
 scan_filepath <- tempfile(fileext = ".nii.gz")
 download.file(url, scan_filepath, mode = "wb")
 nifti <- import_scan(scan_filepath)
url2 <- "https://github.com/Telfer/BoneDensityMapping/releases/download/v1.0.2/test_CT_femur.stl"
 bone_filepath <- tempfile(fileext = ".stl")
 download.file(url2, bone_filepath, mode = "wb")
 surface_mesh <- import_mesh(bone_filepath)

 # get density of surface bone directly
 mat_peak <- voxel_point_intersect(surface_mesh, nifti,
                                   ct_eqn = "linear",
                                   ct_params = c(68.4, 1.106),
                                   check_in_vol = FALSE)

 # remap and get density (for group level comparisons)
 landmark_path <- system.file("extdata", "test_femur.mrk.json",
                              package = "BoneDensityMapping")
 landmarks <- import_lmks(landmark_path)
 mapped_coords <- surface_points_template(surface_mesh, landmarks,
                                          no_surface_sliders = 1000)
 mat_peak <- voxel_point_intersect(mapped_coords, nifti,
                                   ct_eqn = "linear",
                                   ct_params = c(68.4, 1.106),
                                   check_in_vol = FALSE)
```

Index