# Package 'AvInertia'

October 12, 2022

**Title** Calculate the Inertial Properties of a Flying Bird

**Version** 0.0.2

**Description** Tools to compute the center of gravity and moment of inertia tensor of any flying bird. The tools function by modeling a bird as a composite structure of simple geometric objects. This requires detailed morphological measurements of bird specimens although those obtained for the associated paper have been included in the package for use. Refer to the vignettes and supplementary material for detailed information on the package function.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1.9000

**Imports** pracma, readxl, ggplot2, stringr, tidyr, ggthemes, reshape2

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**Depends** R (>= 3.5.0)

**URL** https://github.com/charvey23/AvInertia

**BugReports** https://github.com/charvey23/AvInertia/issues

**NeedsCompilation** no

**Author** Christina Harvey [aut, cre] (<https://orcid.org/0000-0002-2830-0844>), Vikram B. Baliga [aut] (<https://orcid.org/0000-0002-9367-8974>), Jasmin C.M. Wong [aut] (<https://orcid.org/0000-0003-4648-2503>)

**Maintainer** Christina Harvey <harveyca@umich.edu>

**Repository** CRAN

**Date/Publication** 2022-03-24 15:50:02 UTC

# R topics documented:

`calc_inertia_conesolid` *Moment of inertia tensor of a solid circular cone pyramid*

## Description

All outputs are based on an origin at the centered point on the base Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf

## Usage

```
calc_inertia_conesolid(r, h, m)
```

## Arguments

| | |
|---|---|
| `r` | radius of the cone base (m) |
| `h` | height of the cone (m) |
| `m` | mass of the cone (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a solid circular cone about its center of gravity with z oriented through it's major axis.

## CAUTION

Origin of the output tensor is NOT at the center of gravity but at the center of the base.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
```

```
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`calc_inertia_cylhollow` *Moment of inertia tensor of a hollow cylinder*

---

## Description

Reference:https://apps.dtic.mil/sti/pdfs/AD0274936.pdf

## Usage

```
calc_inertia_cylhollow(r_out, r_in, h, m)
```

## Arguments

| | |
|---|---|
| `r_out` | outer radius of the cylinder (m) |
| `r_in` | inner radius of the cylinder (m) |
| `h` | height of the cylinder (m) |
| `m` | mass of the cylinder (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a hollow cylinder about its center of gravity with z oriented through it's major axis

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

# calc_inertia_cylsolid *Moment of inertia tensor of a solid cylinder*

---

## Description

Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf

## Usage

```
calc_inertia_cylsolid(r, h, m)
```

## Arguments

| | |
|---|---|
| r | radius of the cylinder (m) |
| h | height of the cylinder (m) |
| m | mass of the cylinder (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a solid cylinder about its center of gravity with z oriented through it's major axis

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`calc_inertia_ellcone` *Moment of inertia tensor of a solid elliptical cone - end of purple notebook derivation verified in green*

---

## Description

Moment of inertia tensor of a solid elliptical cone - end of purple notebook derivation verified in green

## Usage

```
calc_inertia_ellcone(A, B, l, m)
```

## Arguments

| | |
|---|---|
| A | half height of the wide base (m) |
| B | half width of the wide base (m) |
| l | length of the cone (m) |
| m | mass of the cone (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a solid elliptical cone about the center of the wider base with z oriented towards the end.

## CAUTION

Origin of the output tensor is NOT at the center of gravity but at the center of the base.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
```

```
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`calc_inertia_ellcyl` *Moment of inertia tensor of a solid elliptical cylinder Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf*

---

## Description

Moment of inertia tensor of a solid elliptical cylinder Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf

## Usage

```
calc_inertia_ellcyl(a, b, l, m)
```

## Arguments

| | |
|---|---|
| `a` | half height of the base - oriented along the x axis in torso FOR (z axis in full bird FOR) (m) |
| `b` | half width of the base - oriented along the y axis in torso FOR (y axis in full bird FOR) (m) |
| `l` | length of the cylinder - oriented along the z axis in torso FOR (x axis in full bird FOR) (m) |
| `m` | mass of the cylinder (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a solid elliptical cylinder about it's center of gravity

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

# calc_inertia_ellipse *Moment of inertia tensor of solid ellipse CG or a half ellipse centered on the base*

---

## Description

Reference:https://apps.dtic.mil/sti/pdfs/AD0274936.pdf

## Usage

```
calc_inertia_ellipse(a, b, c, m)
```

## Arguments

| | |
|---|---|
| a | half the height along the x direction (m) |

| | |
|---|---|
| b | half the width along the y direction (m) |
| c | half the length along the z direction (m) |
| m | mass of the ellipse (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a solid ellipse about its center of gravity with the major axes aligned.

## CAUTION

Origin is at the center of gravity for a full ellipse or at the center of the base if modeling a half ellipse.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

calc_inertia_platerect *Moment of inertia tensor of a flat rectangular plate - assumes thickness is approximately zero Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf*

## Description

Moment of inertia tensor of a flat rectangular plate - assumes thickness is approximately zero Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf

## Usage

```
calc_inertia_platerect(w, h, m)
```

## Arguments

| | |
|---|---|
| w | full width of one side of the plate - short edge (m) |
| h | height of the plate - long edge (m) |
| m | mass of the plate (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a flat plate about its center of gravity with z oriented parallel with it's long edge (h) and y along its short edge (w)

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
```

```
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`calc_inertia_platetri` *Moment of inertia tensor of a flat triangular plate*

---

## Description

Reference: https://apps.dtic.mil/dtic/tr/fulltext/u2/a183444.pdf page 4 equations 2.16-2.20

## Usage

```
calc_inertia_platetri(pts, A, rho, t, desired_prop)
```

## Arguments

| | |
|---|---|
| `pts` | a matrix of the three 3D points that define a point on the triangular plate. Frame of reference: Muscle \| Origin: VRP each point should be a different row as follows: pt1x, pt1y, pt1z pt2x, pt1y, pt2z pt3x, pt3y, pt3z |
| `A` | area of the triangular plate (m) |
| `rho` | density of the material (kg/m^3) |
| `t` | thickness of the plate (m) |
| `desired_prop` | a string containing either "I" or "CG" depending on the desired output |

## Value

a 3x3 matrix representing the moment of inertia tensor of a flat triangular plate about its center of gravity. Z axis defined as the normal to the input points.

## Warning

The input points should be defined in a counterclockwise direction around the plate in the triangular plate frame of reference i.e. all z components should be equal

### Author(s)

Christina Harvey

### Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

## calc_inertia_pyrasolid

*Moment of inertia tensor of a solid square pyramid Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf All outputs are based on an origin at the centered point on the base*

---

### Description

Moment of inertia tensor of a solid square pyramid Reference: https://apps.dtic.mil/sti/pdfs/AD0274936.pdf All outputs are based on an origin at the centered point on the base

## Usage

```
calc_inertia_pyrasolid(w, h, m)
```

## Arguments

| | |
|---|---|
| w | entire width of one side of the pyramid base (m) |
| h | entire height of the pyramid (m) |
| m | mass of the pyramid (kg) |

## Value

a 3x3 matrix representing the moment of inertia tensor of a solid square pyramid about its center of gravity with z oriented through it's major axis.

## CAUTION

Origin is NOT at the center of gravity but at the center of the base.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
```

```
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`calc_rot` *A 3x3 rotation matrix constructed by projecting the new axes onto the original system. Likely results in rotation about all axes.*

---

## Description

A 3x3 rotation matrix constructed by projecting the new axes onto the original system. Likely results in rotation about all axes.

## Usage

```
calc_rot(z_vector, x_vector)
```

## Arguments

| | |
|---|---|
| `z_vector` | a 1x3 vector representing the direction for the desired z axis of the new frame of reference. Frame of reference: VRP |
| `x_vector` | a 1x3 vector representing the direction for the desired x axis of the new frame of reference. Frame of reference: VRP |

## Value

a 3x3 matrix representing the rotation matrix that transforms between VRP frame and object frame

## Author(s)

Christina Harvey

## Examples

```
library(AvInertia)
z_vector = c(0,0,1)
x_vector = c(-1,0,0)

# should return matrix [[-1,0,0];[0,-1,0];[0,0,1]]
calc_rot(z_vector, x_vector)
```

## calc_univec *Determine the unit vector of any input vector*

### Description

Determine the unit vector of any input vector

### Usage

```
calc_univec(vector)
```

### Arguments

| | |
|---|---|
| vector | any vector or array with only one dimension |

### Value

the unit vector in the size of the input vector

### Author(s)

Christina Harvey

### Examples

```
library(AvInertia)

#any random input vector
vector = c(1,2,3)
output_vec = calc_univec(vector)

# if unit vector the magnitude should = 1
pracma::Norm(output_vec)
```

## clean_pts *The identified peripheral and joint 3D positions.*

### Description

A dataset containing the identified peripheral and joint 3D positions for a pigeon BirdID = 20_0300, TestID = 20_03004, FrameID = 317

### Usage

```
clean_pts
```

### Format

A matrix with 10 rows and 3 columns. Columns represent x, y, and z coordinate respectively:

**pt1x, pt1y, pt1z** Point on the shoulder joint (m).

**pt2x, pt1y, pt2z** Point on the elbow joint (m).

**pt3x, pt3y, pt3z** Point on the wrist joint (m).

**pt4x, pt4y, pt4z** Point on the end of carpometacarpus (m).

**pt6x, pt6y, pt6z** Point on the leading edge of the wing in front of the wrist joint (m).

**pt8x, pt8y, pt8z** Point on tip of most distal primary (m).

**pt9x, pt9y, pt9z** Point on the tip of the last primary to model as if it is on the end of the carpometacarpus (m).

**pt10x, pt10y, pt10z** Point on tip of last primary to model as if it was distributed along the carpometacarpus (m).

**pt11x, pt11y, pt11z** Point on tip of most proximal feather (m).

**pt12x, pt12y, pt12z** Point on exterior shoulder position (wing root leading edge) (m).

---

## combine_inertialprop *Combine body and wing inertial components.*

---

### Description

Combines data exported from massprop_restbody and massprop_birdwing.

### Usage

```
combine_inertialprop(
  curr_torsotail_data,
  left_wing_data,
  right_wing_data,
  dat_id_curr,
  dat_bird_curr,
  symmetric
)
```

### Arguments

`curr_torsotail_data` Output dataframe from massprop_restbody.

`left_wing_data` Output dataframe from massprop_birdwing.

`right_wing_data` Output dataframe from massprop_birdwing.

`dat_id_curr` Dataframe related to the current bird wing ID info that must include the following columns:

- speciesSpecies ID code as a string.
- BirdIDBird ID code as a string.
- TestIDTest ID code as a string.
- frameIDVideo frame ID code as a string.

`dat_bird_curr` Dataframe related to the current bird wing that must include the following columns:

- total_bird_massMass of full bird for the current wing (kg).

`symmetric` Logical indicating if the input wings are symmetric or not. If True than left_wing_data = right_wing_data.

## Value

a dataframe containing all of the inertial properties for each wing component and the full bird about it's center of gravity and the vehicle reference point (VRP)

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

`compute_feat_inertia` *Compute the inertia of the individual feathers*

## Description

Compute the inertia of the individual feathers

## Usage

```
compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
```

## Arguments

`dat_mat` Dataframe related to the current species input as a dataframe with the following structure:

- materialMaterial information. Must include the following: "Cortex", "Medullary"
- densityDensity of each material (kg/m^3)

`dat_feat_curr` Dataframe related to the current bird wing feathers input as a dataframe with the following structure:

- featherFeather ID code. Must be in standard format i.e. 1st primary is "P1", third secondary is "S3", etc. Alula feathers should be grouped and named "alula".
- m_fMass of feather in the same row as the appropriate feather ID code (kg)
- l_calLength of calamus in the same row as the appropriate feather ID code (m)
- l_vaneLength of rachis/vane in the same row as the appropriate feather ID code (m)
- w_calWidth (diameter) of calamus in the same row as the appropriate feather ID code (m)
- w_vpWidth of proximal vane (average value) in the same row as the appropriate feather ID code (m)
- w_vdWidth of distal vane (average value) in the same row as the appropriate feather ID code (m)
- vane_angleInterior angle between the rachis and calamus (degrees)

NOTE: Alula feathers will be treated as point mass so only the mass of the feathers is required. Other columns can be left blank.

`dat_bird_curr` Dataframe related to the current bird wing that must include the following columns:

- barb_radiusRadius of feather barb for current species (m)
- barb_distanceDistance between feather barbs for current species (m)

## Value

A list with one entry per flight feather. Each primary feather includes the following variables:

- I_pria 3x3 matrix representing the moment of inertia about each feather calamus tip (kg-m^2).
- CG_pria 1x3 vector (x,y,z) representing the center of gravity of the primary feather (m).
- m_pria double representing the mass of the primary feather (kg).

Each secondary feather includes the following variables:

- I_seca 3x3 matrix representing the moment of inertia about each feather calamus tip (kg-m^2).
- CG_seca 1x3 vector (x,y,z) representing the center of gravity of the primary feather (m).
- m_seca double representing the mass of the primary feather (kg).

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

dat_bird_curr *Bird specific morphology dataset*

## Description

A dataset containing the bird specific morphology data for a pigeon BirdID = 20_0300, TestID = 20_03004, FrameID = 317

## Usage

```
dat_bird_curr
```

## Format

A dataframe with 1 row and 125 variables. Only 29 required for proper operation:

**total_bird_mass** Mass of full bird for the current wing (kg)

**wing_mass** Mass of one wing, should be the current wing (kg)

**barb_radius** Radius of feather barb for current species (m)

**barb_distance** Distance between feather barbs for current species (m)

**brachial_muscle_mass** Mass of all muscles in the brachial region of the wing (kg)

**antebrachial_muscle_mass** Mass of all muscles in the antebrachial region of the wing (kg)

**manus_muscle_mass** Mass of all muscles in the manus region of the wing (kg)

**all_skin_coverts_mass** Mass of all skin and covert feathers (kg).

**tertiary_mass** Mass of tertiary feathers (kg).

**extend_neck** Logical input defining whether the neck should be modeled as extended or not

**head_length** Length of the head from base to tip (m)

**head_mass** Mass of the head (kg)

**head_height** Height of the head at the base (m)

**neck_mass** Mass of the neck (kg)

**neck_width** OPTIONAL: Average width of the stretched neck (m)

**neck_length** OPTIONAL: Length of the stretched neck (m)

**torsotail_length** Length from the beginning of the torso to the tip of the tail (m)

**torsotail_mass** Mass of the torso and tail (kg)

**tail_length** Length of the tail (m)

**tail_mass** Mass of the tail (kg)

**tail_width** Average width of the furled tail (m)

**right_leg_mass** Mass of the right leg (kg)

**left_leg_mass** Mass of the left leg (kg)

**body_width_max** Maximum width of the torso (m)

**body_width_at_leg_insert** Width of the body at the point where the legs are inserted (m).

**x_loc_of_body_max** x coordinate from the VRP in the full bird frame of reference of the maximum body width (m).

**x_loc_leg_insertion** x coordinate from the VRP in the full bird frame of reference of the leg insertion location (m).

**x_loc_TorsotailCoG** x coordinate from the VRP in the full bird frame of reference of the center of gravity of the torso and tail (m).

**z_loc_TorsotailCoG** x coordinate from the VRP in the full bird frame of reference of the the center of gravity of the torso and tail (m).

---

`dat_bone_curr` *Wing bone specific morphology dataset*

---

## Description

A dataset containing the wing bone specific data for a pigeon BirdID = 20_0300, TestID = 20_03004, FrameID = 317

## Usage

```
dat_bone_curr
```

## Format

A dataframe with 6 rows and 5 variables:

**bone** Bone ID code. Must include: "Humerus","Ulna","Radius","Carpometacarpus","Ulnare" and "Radiale".

**bone_mass** Mass of bone in the same row as the appropriate bone ID code (kg)

**bone_len** Length of bone in the same row as the appropriate bone ID code (m)

**bone_out_rad** Outer radius of bone in the same row as the appropriate bone ID code (m)

**bone_in_rad** Inner radius of bone in the same row as the appropriate bone ID code (m)

## dat_feat_curr *Flight feather specific morphology dataset*

### Description

A dataset containing the wing bone specific data for a pigeon BirdID = 20_0300, TestID = 20_03004, FrameID = 317

### Usage

```
dat_feat_curr
```

### Format

A dataframe with 20 rows and 15 variables. Only 8 variables required for proper functioning:

**feather** Feather ID code. Must be in standard format i.e. 1st primary is "P1", third secondary is "S3", etc. Alula feathers should be grouped and named "alula".

**m_f** Mass of feather in the same row as the appropriate feather ID code (kg)

**l_cal** Length of calamus in the same row as the appropriate feather ID code (m)

**l_vane** Length of rachis/vane in the same row as the appropriate feather ID code (m)

**w_cal** Width (diameter) of calamus in the same row as the appropriate feather ID code (m)

**w_vp** Width of proximal vane (average value) in the same row as the appropriate feather ID code (m)

**w_vd** Width of distal vane (average value) in the same row as the appropriate feather ID code (m)

**vane_angle** Interior angle between the rachis and calamus (degrees)

## dat_id_curr *Identification variables for current configuration*

### Description

A dataset containing the identification data for a pigeon BirdID = 20_0300, TestID = 20_03004, FrameID = 317

### Usage

```
dat_id_curr
```

### Format

A dataframe with 1 row and 4 required variables:

**species** Species ID code as a string

**BirdID** Bird ID code as a string

**TestID** Test ID code as a string

**frameID** Video frame ID code as a string

---

## dat_mat *Material properties.*

---

### Description

A dataset containing the material properties for a pigeon BirdID = 20_0300, TestID = 20_03004, FrameID = 317.

### Usage

```
dat_mat
```

### Format

A dataframe with 20 rows and 15 variables. Only 8 variables required for proper functioning:

**material** Material information. Must include the following: "Bone","Skin","Muscle","Cortex", "Medullary"

**density** Density of each material (kg/m^3)

---

## density_optimizer *Optimize torso section densities*

---

### Description

Function that is used within an optimization routine to select the appropriate torso section density

### Usage

```
density_optimizer(x, m_body, A, v_ell, CG_body_x, CG_ell, rho_avg)
```

## Arguments

| | |
|---|---|
| x | a scalar guess at the density of the torso hemi-ellipsoid section (kg/m^3) |
| m_body | a scalar representing the mass of the full torso |
| A | a 2x2 matrix representing the mass and volume calculations of the final two torso section |
| v_ell | a scalar representing the volume of the hemi-ellipsoid |
| CG_body_x | a scalar representing the position of the center of gravity of the full torso along the x axis with the origin at the VRP |
| CG_ell | a scalar representing the position of the center of gravity of the hemi-ellipsoid section along the x axis with the origin at the VRP |
| rho_avg | average density of the full torso |

## Value

the squared error between the three section densities and the average torso density

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

`kronecker_delta` *Kroneckerdelta function*

## Description

Kroneckerdelta function

## Usage

```
kronecker_delta(i, j)
```

## Arguments

| | |
|---|---|
| `i` | a scalar index (usually row of a matrix) |
| `j` | a scalar index (usually column of a matrix) |

## Value

a scalar value. Returns 1 if i and j are equal otherwise returns 0

## Author(s)

Christina Harvey

## Examples

```
library(AvInertia)

# should return 1
kronecker_delta(1,1)

# should return 0
kronecker_delta(0,1)
```

`massprop_birdwing` *Calculate the center of gravity and moment of inertia for a halfspan wing.*

## Description

Function that reads in anatomical data and returns the moment of inertia tensor and center of gravity of a wing one side of the bird.

## Usage

```
massprop_birdwing(
  dat_wingID_curr,
  dat_bird_curr,
  dat_bone_curr,
  dat_feat_curr,
  dat_mat_curr,
  clean_pts,
  feather_inertia,
  plot_var
)
```

## Arguments

`dat_wingID_curr`
: Dataframe related to the current bird wing ID info that must include the following columns:
  - speciesSpecies ID code as a string.
  - BirdIDBird ID code as a string.
  - TestIDTest ID code as a string.
  - frameIDVideo frame ID code as a string.

`dat_bird_curr`
: Dataframe related to the current bird wing that must include the following columns:
  - total_bird_massMass of full bird for the current wing (kg).
  - wing_massMass of one wing, should be the current wing (kg).
  - barb_radiusRadius of feather barb for current species (m).
  - barb_distanceDistance between feather barbs for current species (m).
  - brachial_muscle_massMass of all muscles in the brachial region of the wing (kg).
  - antebrachial_muscle_massMass of all muscles in the antebrachial region of the wing (kg).
  - manus_muscle_massMass of all muscles in the manus region of the wing (kg).
  - all_skin_coverts_massMass of all skin and covert feathers (kg).
  - tertiary_massMass of tertiary feathers (kg).

`dat_bone_curr`
: Dataframe (6 row x 5 column) related to the current bird wing bones that must include the following columns:
  - boneBone ID code. Must include: "Humerus","Ulna","Radius","Carpometacarpus","Ulnare" and "Radiale".
  - bone_massMass of bone in the same row as the appropriate bone ID code (kg).
  - bone_lenLength of bone in the same row as the appropriate bone ID code (m).
  - bone_out_radOuter radius of bone in the same row as the appropriate bone ID code (m).

- bone_in_radInner radius of bone in the same row as the appropriate bone ID code (m).

`dat_feat_curr` Dataframe related to the current bird wing feathers input as a dataframe with the following structure:

- featherFeather ID code. Must be in standard format i.e. 1st primary is "P1", third secondary is "S3", etc. Alula feathers should be grouped and named "alula".
- m_fMass of feather in the same row as the appropriate feather ID code (kg).
- l_calLength of calamus in the same row as the appropriate feather ID code (m).
- l_vaneLength of rachis/vane in the same row as the appropriate feather ID code (m).
- w_calWidth (diameter) of calamus in the same row as the appropriate feather ID code (m).
- w_vpWidth of proximal vane (average value) in the same row as the appropriate feather ID code (m).
- w_vdWidth of distal vane (average value) in the same row as the appropriate feather ID code (m).
- vane_angleInterior angle between the rachis and calamus (degrees).

NOTE: Alula feathers will be treated as point mass so only the mass of the feathers is required. Other columns can be left blank.

`dat_mat_curr` Dataframe related to the current species input as a dataframe with the following structure:

- materialMaterial information. Must include the following: "Bone","Skin","Muscle","Cortex", "Medullary"
- densityDensity of each material (kg/m^3).

`clean_pts` A data frame of the key positions of the bird as follows:

- pt1x, pt1y, pt1zPoint on the shoulder joint (m).
- pt2x, pt1y, pt2zPoint on the elbow joint (m).
- pt3x, pt3y, pt3zPoint on the wrist joint (m).
- pt4x, pt4y, pt4zPoint on the end of carpometacarpus (m).
- pt6x, pt6y, pt6zPoint on the leading edge of the wing in front of the wrist joint (m).
- pt8x, pt8y, pt8zPoint on tip of most distal primary (m).
- pt9x, pt9y, pt9zPoint on the tip of the last primary to model as if it is on the end of the carpometacarpus (m).
- pt10x, pt10y, pt10zPoint on tip of last primary to model as if it was distributed along the carpometacarpus (m).
- pt11x, pt11y, pt11zPoint on tip of most proximal feather (m).
- pt12x, pt12y, pt12zPoint on exterior shoulder position (wing root leading edge) (m).

`feather_inertia`

A list with one entry per flight feather. Each primary feather includes the following variables:

- I_pria 3x3 matrix representing the moment of inertia about each feather calamus tip (kg-m^2).
- CG_pria 1x3 vector (x,y,z) representing the center of gravity of the primary feather (m).
- m_pria double representing the mass of the primary feather (kg).

Each secondary feather includes the following variables:

- I_seca 3x3 matrix representing the moment of inertia about each feather calamus tip (kg-m^2).
- CG_seca 1x3 vector (x,y,z) representing the center of gravity of the primary feather (m).
- m_seca double representing the mass of the primary feather (kg).

`plot_var` A string that defines the x-axis and y-axis of the output plot. Can either equal "yx" or "yz".

## Value

Function returns a dataframe that includes the moment of inertia and center of gravity of one wing about the VRP in the VRP frame and that of each major anatomical group i.e. skin, feathers, bones, muscles.

## CAUTION

All points must all have the vehicle reference point (VRP) as their origin and the vehicle major axes as their frame of reference. This is normally selected so that the VRP is in line with the body center of gravity. Ensure the axes used represent a right-handed axis system.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
```

```
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`massprop_bones` *Bone mass properties*

---

## Description

Calculate the moment of inertia of a bone modeled as a hollow cylinder with two solid end caps

## Usage

```
massprop_bones(m, l, r_out, r_in, rho, start, end)
```

## Arguments

| | |
|---|---|
| `m` | Mass of bone (kg) |
| `l` | Length of bone (kg) |
| `r_out` | Outer radius of bone (m) |
| `r_in` | Inner radius of bone (m) |
| `rho` | Density of the bone (kg/m^3) |
| `start` | a 1x3 vector (x,y,z) representing the 3D point where bone starts. Points from the VRP to the bone start. Frame of reference: VRP \| Origin: VRP |
| `end` | a 1x3 vector (x,y,z) representing the 3D point where bone ends. Points from the VRP to the bone start. Frame of reference: VRP \| Origin: VRP |

## Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a bone modeled as a hollow cylinder with two solid end caps
- CGa 1x3 vector representing the center of gravity position of a bone modeled as a hollow cylinder with two solid end caps
- ma double that returns the input bone mass

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

# massprop_feathers *Feather mass properties*

---

## Description

Calculate the moment of inertia of the feathers within the feather frame of reference.

## Usage

```
massprop_feathers(
  m_f,
  l_c,
  l_r_cor,
  w_cal,
  r_b,
  d_b,
  rho_cor,
  rho_med,
  w_vp,
  w_vd,
  angle
)
```

## Arguments

| | |
|---|---|
| m_f | Mass of the entire feather (kg) |
| l_c | Length of the calamus; start of vane to end of calamus(m) |
| l_r_cor | Length of rachis; tip to start of vane (m) |
| w_cal | Width (diameter) of the cortex part of the calamus (m) |
| r_b | Radius of feather barbs (m) |
| d_b | Distance between barbs (m) |
| rho_cor | Density of the cortex (kg/m^3) |
| rho_med | Density of the medullary (kg/m^3) |
| w_vp | Width of proximal (closest to body) vane (m) |
| w_vd | Width of distal (closest to wing tip) vane (m) |
| angle | Angle between calamus and the vane taken the supplement angle to the interior angle. Negative indicates the feather tip is rotated proximally relative to the start of the feather vane. |

## Value

a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a simplified feather with the origin at the feather calamus end and within the feather frame of reference
- CGa 1x3 vector representing the center of gravity position of a simplified feather with the origin at the feather calamus end and within the feather frame of reference
- ma double that returns the feather mass

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

CAUTION: While computing the variable components of the feather the x axis is the normal of the feather.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

# massprop_head *Head mass properties*

---

## Description

Calculate the moment of inertia of a head modeled as a solid cone

## Usage

```
massprop_head(m, r, l, start, end)
```

### Arguments

| | |
|---|---|
| m | Mass of head (kg) |
| r | Maximum head radius (m) |
| l | Maximum head length (m) |
| start | a 1x3 vector (x,y,z) representing the 3D point where head starts. Frame of reference: VRP \| Origin: VRP |
| end | a 1x3 vector (x,y,z) representing the 3D point where head ends. Frame of reference: VRP \| Origin: VRP |

### Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a head modeled as a solid cone
- CGa 1x3 vector representing the center of gravity position of a head modeled as a solid cone
- ma double that returns the head mass

### Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

### Author(s)

Christina Harvey

---

## massprop_muscles *Muscle mass properties*

---

### Description

Calculate the moment of inertia of a muscle modeled as a solid cylinder distributed along the bone length

### Usage

```
massprop_muscles(m, rho, l, start, end)
```

### Arguments

| | |
|---|---|
| m | Mass of muscle (kg). |
| rho | Density of muscle (kg/m^3). |
| l | Length of the muscle group (m). |
| start | a 1x3 vector (x,y,z) representing the 3D point where bone starts. Frame of reference: VRP \| Origin: VRP. |
| end | a 1x3 vector (x,y,z) representing the 3D point where bone ends. Frame of reference: VRP \| Origin: VRP. |

## Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a muscle modeled as a solid cylinder distributed along the bone length.
- CGa 1x3 vector representing the center of gravity position of a muscle modeled as a solid cylinder distributed along the bone length.
- ma double that returns the input muscle mass.

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

`massprop_neck` *Neck mass properties*

## Description

Calculate the moment of inertia of a neck modeled as a solid cylinder

## Usage

```
massprop_neck(m, r, l, start, end)
```

## Arguments

| | |
|---|---|
| `m` | Mass of muscle (kg). |
| `r` | Radius of the neck (m). |
| `l` | Length of the stretched neck (m). |
| `start` | a 1x3 vector (x,y,z) representing the 3D point where neck starts. Frame of reference: VRP \| Origin: VRP. |
| `end` | a 1x3 vector (x,y,z) representing the 3D point where neck ends. Frame of reference: VRP \| Origin: VRP. |

## Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a neck modeled as a solid cylinder.
- CGa 1x3 vector representing the center of gravity position of a neck modeled as a solid cylinder.
- ma double that returns the neck mass.

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

## Author(s)

Christina Harvey

massprop_pm *Point-mass mass properties*

## Description

Calculate the moment of inertia of any point mass

## Usage

```
massprop_pm(m, pt)
```

## Arguments

| | |
|---|---|
| m | Mass of point mass (kg) |
| pt | a 1x3 vector (x,y,z) representing the location of the point mass. Frame of reference: VRP \| Origin: VRP |

## Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a point mass
- CGa 1x3 vector representing the center of gravity position of a point mass
- ma double that returns the input mass

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
```

```
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`massprop_restbody` *Calculate the center of gravity and moment of inertia for the head, neck, torso and tail.*

---

## Description

Function that reads in anatomical data and returns the moment of inertia tensor and center of gravity for the head, neck, tail and torso.

## Usage

```
massprop_restbody(dat_wingID_curr, dat_bird_curr)
```

## Arguments

`dat_wingID_curr`

Dataframe related to the current bird wing ID info that must include the following columns:

- speciesSpecies ID code as a string.
- BirdIDBird ID code as a string.
- TestIDTest ID code as a string.
- frameIDVideo frame ID code as a string.

`dat_bird_curr` Dataframe related to the current bird wing that must include the following columns:

- extend_neckLogical input defining whether the neck should be modeled as extended or not.
- head_lengthLength of the head from base to tip (m).

- head_massMass of the head (kg).
- head_heightHeight of the head at the base (m).
- neck_massMass of the neck (kg).
- neck_widthOPTIONAL - Average width of the stretched neck (m).
- neck_lengthOPTIONAL - Length of the stretched neck (m).
- torsotail_lengthLength from the beginning of the torso to the tip of the tail (m).
- torsotail_massMass of the torso and tail (kg).
- tail_lengthLength of the tail (m).
- tail_massMass of the tail (kg).
- tail_widthAverage width of the furled tail (m).
- right_leg_massMass of the right leg (kg).
- left_leg_massMass of the left leg (kg).
- body_width_maxMaximum width of the torso (m).
- body_width_at_leg_insertWidth of the body at the point where the legs are inserted (m).
- x_loc_of_body_maxx coordinate from the VRP in the full bird frame of reference of the maximum body width (m).
- x_loc_leg_insertionx coordinate from the VRP in the full bird frame of reference of the leg insertion location (m).
- x_loc_TorsotailCoGx coordinate from the VRP in the full bird frame of reference of the center of gravity of the torso and tail (m).
- z_loc_TorsotailCoGx coordinate from the VRP in the full bird frame of reference of the the center of gravity of the torso and tail (m).

## Value

Function returns a dataframe that includes the moment of inertia and center of gravity of head, neck, torso and tail.

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
```

```
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`massprop_skin` *Calculates the mass properties of skin or tertiaries*

---

## Description

Calculate the moment of inertia of skin or tertiaries modeled as a flat triangular plate

## Usage

```
massprop_skin(m, rho, pts)
```

## Arguments

| | |
|---|---|
| `m` | Mass of skin (kg) |
| `rho` | Density of skin (kg/m^3) |
| `pts` | a 3x3 matrix that represent three points that define the vertices of the triangle. Frame of reference: VRP \| Origin: VRP Must be numbered in a counterclockwise direction for positive area, otherwise signs will be reversed. each point should be a different of the matrix as follows:<br>• pt1x, pt1y, pt1z<br>• pt2x, pt1y, pt2z<br>• pt3x, pt3y, pt3z |

## Value

This function returns a list that includes: point mass

- Ia 3x3 matrix representing the moment of inertia tensor of skin modeled as a flat triangular plate
- CGa 1x3 vector representing the center of gravity position of skin modeled as a flat triangular plate
- ma double that returns the input skin mass

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

Caution: The skin frame of reference assumes that the z axis is normal to the incoming points.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`massprop_tail` *Head mass properties*

---

## Description

Calculate the moment of inertia of a tail modeled as a solid cone

## Usage

```
massprop_tail(m, l, w, start, end)
```

## Arguments

| | |
|---|---|
| m | Mass of muscle (kg) |
| l | Length of the tail (m) |
| w | Width of the tail (m) |
| start | a 1x3 vector (x,y,z) representing the 3D point where head starts. Frame of reference: VRP \| Origin: VRP |
| end | a 1x3 vector (x,y,z) representing the 3D point where head ends. Frame of reference: VRP \| Origin: VRP |

## Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a head modeled as a solid cone
- CGa 1x3 vector representing the center of gravity position of a head modeled as a solid cone
- ma double that returns the head mass

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
```

```
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`massprop_torso` *Torso and leg mass properties*

---

## Description

Calculate the moment of inertia of a head modeled as a solid cone

## Usage

```
massprop_torso(
  m_true,
  m_legs,
  w_max,
  h_max,
  l_bmax,
  w_leg,
  l_leg,
  l_tot,
  CG_true_x,
  CG_true_z,
  start,
  end
)
```

## Arguments

| | |
|---|---|
| `m_true` | Mass of the torso and legs - no tail (kg) |
| `m_legs` | Mass of the legs only (kg) |
| `w_max` | Maximum width of the body (m) |
| `h_max` | Maximum height of the body (m) |
| `l_bmax` | x location of the maximum width of the body (m) |

| | |
|---|---|
| w_leg | width of the body at leg insertion (m) |
| l_leg | x location of the leg insertion point (m) |
| l_tot | length of body from clavicle to beginning of the tail (m) |
| CG_true_x | x location of the CG for the torso and legs, origin is at the VRP, measured positive if aft of the VRP (m) |
| CG_true_z | z location of the CG for the torso and legs, origin is at the VRP (m) |
| start | a 1x3 vector (x,y,z) representing the 3D point where torso starts. Frame of reference: VRP \| Origin: VRP |
| end | a 1x3 vector (x,y,z) representing the 3D point where tail ends. Frame of reference: VRP \| Origin: VRP |

## Value

This function returns a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of the torso and leg composite body
- CGa 1x3 vector representing the center of gravity position of the torso and leg composite body
- ma double that returns the mass of the torso and leg composite body

## Warning

Parallel axis theorem does not apply between two arbitrary points. One point must be the object's center of gravity.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
```

```
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`orient_feather` *Determine the feather orientation*

---

## Description

Code that returns the orientation of each primary and secondary feather on the wing.

## Usage

```
orient_feather(no_pri, no_sec, Pt1, Pt2, Pt3, Pt4, Pt8, Pt9, Pt10, Pt11, Pt12)
```

## Arguments

| | |
|---|---|
| `no_pri` | a scalar representing the amount of primary feathers. |
| `no_sec` | a scalar representing the amount of secondary feathers. |
| `Pt1` | a 1x3 vector (x,y,z) representing the point on the shoulder joint (m). |
| `Pt2` | a 1x3 vector (x,y,z) representing the point on the elbow joint (m). |
| `Pt3` | a 1x3 vector (x,y,z) representing the point on the wrist joint (m). |
| `Pt4` | a 1x3 vector (x,y,z) representing the point on the end of carpometacarpus (m). |
| `Pt8` | a 1x3 vector (x,y,z) representing the point on tip of most distal primary (m). |
| `Pt9` | a 1x3 vector (x,y,z) representing the point on the tip of the last primary to model as if it is on the end of the carpometacarpus (m). |
| `Pt10` | 1x3 vector (x,y,z) representing the point on tip of last primary to model as if it was distributed along the carpometacarpus (m). Usually the first secondary feather tip. |
| `Pt11` | 1x3 vector (x,y,z) representing the point on tip of most proximal secondary feather (m). |
| `Pt12` | 1x3 vector (x,y,z) representing the point on exterior shoulder position (wing root leading edge) (m). |

## Value

a list called "feather". This contains three matrices.

1. "loc_start" a matrix defining the 3D point where each feather starts. Rows are the different feathers and columns are x, y, z coordinates respectively.
2. "loc_end" a matrix defining the 3D point where each feather end. Rows are the different feathers and columns are x, y, z coordinates respectively.
3. "normal" a matrix that gives the vector that defines the normal to each feather plane. Rows are the different feathers and columns are x, y, z vector directions respectively.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

parallelaxis *Parallel axis theory*

## Description

Reads in an initial tensor and an offset to compute the transformed tensor. Will be in the same frame of reference as the input tensor.

## Usage

```
parallelaxis(I, offset_vec, m, cg_a)
```

## Arguments

| | |
|---|---|
| I | a 3x3 matrix representing the moment of inertia tensor about the center of gravity of the object (kg-m^2). |
| offset_vec | a 1x3 vector representing the distance (x,y,z) between the objects CG and the arbitrary pt A (m). Vector should always point from the CG to the arbitrary point A. |
| m | Mass of the object (kg). |
| cg_a | If input I is about the CG enter "CG" or if I is about an arbitrary axis enter "A". |

## Value

a 3x3 matrix representing the transformed moment of inertia tensor after a solid body translation defined by the offset vector.

## Author(s)

Christina Harvey

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
```

```
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`plot_CGloc` *Plot the center of gravity of each component*

---

## Description

Plot the center of gravity of each component

## Usage

```
plot_CGloc(
  clean_pts,
  mass_properties,
  mass_properties_skin,
  mass_properties_bone,
  mass_properties_feathers,
  mass_properties_muscle,
  prop_tertiary1,
  prop_tertiary2,
  plot_var
)
```

## Arguments

`clean_pts` Dataframe of the key positions of the bird as follows:

- pt1x, pt1y, pt1zPoint on the shoulder joint
- pt2x, pt1y, pt2zPoint on the elbow joint
- pt3x, pt3y, pt3zPoint on the wrist joint
- pt4x, pt4y, pt4zPoint on the end of carpometacarpus
- pt6x, pt6y, pt6zPoint on the leading edge of the wing in front of the wrist joint

- pt8x, pt8y, pt8zPoint on tip of most distal primary
- pt9x, pt9y, pt9zPoint that defines the end of carpometacarpus
- pt10x, pt10y, pt10zPoint on tip of last primary to model as if on the end of the carpometacarpus
- pt11x, pt11y, pt11zPoint on tip of most proximal feather (wing root trailing edge)
- pt12x, pt12y, pt12zPoint on exterior shoulder position (wing root leading edge)

`mass_properties`
: Dataframe containing the center of gravity and moment of inertia components of the full wing.

`mass_properties_skin`
: Dataframe containing the center of gravity and moment of inertia components of the skin. Formatted with the following columns: "species","BirdID","TestID","FrameID", "prop_type","component","value".

`mass_properties_bone`
: Dataframe containing the center of gravity and moment of inertia components of the wing bones. Formatted with the following columns: "species","BirdID","TestID","FrameID", "prop_type","component","value".

`mass_properties_feathers`
: Dataframe containing the center of gravity and moment of inertia components of the feathers. Formatted with the following columns: "species","BirdID","TestID","FrameID", "prop_type","component","value".

`mass_properties_muscle`
: Dataframe containing the center of gravity and moment of inertia components of the muscles Formatted with the following columns: "species","BirdID","TestID","FrameID", "prop_type","component","value".

`prop_tertiary1` Dataframe containing the center of gravity and moment of inertia components of the first tertiary group. Formatted with the following columns: "species","BirdID","TestID","FrameID", "prop_type","component","value".

`prop_tertiary2` Dataframe containing the center of gravity and moment of inertia components of the second tertiary group. Formatted with the following columns: "species","BirdID","TestID","FrameID' "prop_type","component","value".

`plot_var` Character of either "yx" or "yz". Defines the output axes of the plot.

## Value

A plot of the request axes

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
```

```
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

`rotx` *A 3x3 rotation matrix allowing rotation about the x-axis. Constructed using a cosine rotation matrix where the rotation angle in degrees is measured counterclockwise allowing positive rotation under the right hand rule.*

---

## Description

A 3x3 rotation matrix allowing rotation about the x-axis. Constructed using a cosine rotation matrix where the rotation angle in degrees is measured counterclockwise allowing positive rotation under the right hand rule.

## Usage

```
rotx(angle)
```

## Arguments

| | |
|---|---|
| `angle` | a scalar representing the angle to rotate (degrees) |

## Value

a 3x3 matrix representing the rotation about the x-axis by the given angle

### Author(s)

Christina Harvey

### Examples

```
library(AvInertia)
angle = 90
# should return matrix [[1,0,0];[0,0,1];[0,1,0]]
rotx(angle)
```

---

## store_data *Store data from the inertia calculations in long format*

---

### Description

Function to store moment of inertia tensor and center of gravity vector components in long format

### Usage

```
store_data(dat_wingID_curr, dat_mass, mass_properties, name)
```

### Arguments

| | |
|---|---|
| dat_wingID_curr | Dataframe related to the current bird wing ID info that must include the following columns:<br>• speciesSpecies ID code as a string<br>• BirdIDBird ID code as a string<br>• TestIDTest ID code as a string<br>• frameIDVideo frame ID code as a string |
| dat_mass | Dataframe containing the new MOI and CG data to add to mass_properties as new rows. Must include:<br>• IMoment of inertia tensor (kg-m^2)<br>• CGCenter of gravity with three location components (m) |
| mass_properties | Dataframe containing any previously saved data. Must have the following columns: "species","BirdID","TestID","FrameID", "prop_type","component","value". |
| name | Name of the component for which the moment of inertia and center of gravity were computed. |

### Value

This function returns mass_properties as an updated dataframe with a new row corresponding to the dat_mass information

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

---

# structural2VRP_feat *Transform feather specific inertial properties to current position*

---

## Description

Transform feather specific inertial properties to current position

## Usage

```
structural2VRP_feat(m_f, I_fCG, CG_start, start, end, normal)
```

## Arguments

| | |
|---|---|
| m_f | a scalar representing the mass of the feather (kg) |

| | |
|---|---|
| I_fCG | a 3x3 matrix representing the moment of inertia tensor with the origin at the feather calamus end and within the feather frame of reference |
| CG_start | a 1x3 vector representing the center of gravity of the feather with the origin at the feather calamus end and within the feather frame of reference |
| start | a 1x3 vector representing the location of the feather calamus end with the origin at the VRP and within the full bird frame of reference |
| end | a 1x3 vector representing the location of the feather tip with the origin at the VRP and within the full bird frame of reference |
| normal | a 1x3 vector representing the normal to the plane of the feather vanes within the full bird frame of reference |

## Value

a list that includes:

- Ia 3x3 matrix representing the moment of inertia tensor of a simplified feather with the origin at the VRP and within the full bird frame of reference
- CGa 1x3 vector representing the center of gravity position of a simplified feather with the origin at the VRP and within the full bird frame of reference
- ma double that returns the feather mass

## Examples

```
# refer to the vignette
library(AvInertia)

# load data
data(dat_id_curr, package = "AvInertia")
data(dat_bird_curr, package = "AvInertia")
data(dat_feat_curr, package = "AvInertia")
data(dat_bone_curr, package = "AvInertia")
data(dat_mat, package = "AvInertia")
data(clean_pts, package = "AvInertia")

# 1. Determine the center of gravity of the bird's torso (including the legs)
dat_torsotail_out = massprop_restbody(dat_id_curr, dat_bird_curr)
# 2. Calculate the inertia of the flight feathers about the tip of the calamus
feather_inertia <- compute_feat_inertia(dat_mat, dat_feat_curr, dat_bird_curr)
# 3. Determine the center of gravity of one of the bird's wings
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = 0)
# Visualize the center of gravity of each wing component in the x and y axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yx")
# or the y and z axis
dat_wing_out      = massprop_birdwing(dat_id_curr, dat_bird_curr,
dat_bone_curr, dat_feat_curr, dat_mat, clean_pts,
feather_inertia, plot_var = "yz")
```

```
# 4. Combine all data and obtain the center of gravity, moment of inertia
# and principal axes of the bird
curr_full_bird      = combine_inertialprop(dat_torsotail_out,dat_wing_out,
dat_wing_out, dat_id_curr, dat_bird_curr, symmetric=TRUE)
```

# Index