mecheng

1.0.0

Generated by Doxygen 1.8.11

Contents

1 Modules Index

1.1 Modules List

Here is a list of all documented modules with brief descriptions:

2 Data Type Index

2.1 Data Types List

Here are the data types with brief descriptions:

3 Module Documentation

3.1 columns Module Reference

columns

Data Types

- interface buckling_load

  Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column.
- interface euler_buckling

  Computes the critical buckling load of a column by means of Euler's equation.
- interface johnson_buckling

  Computes the critical buckling load of a column by means of J.B. Johnson's equation.
- interface slenderness_ratio

  Computes the slenderness ratio of a column.

Functions/Subroutines

- pure elemental real(real64) function slenderness_ratio_1 (length, rad)

  Computes the slenderness ratio of a column.
- pure elemental real(real64) function slenderness_ratio_2 (length, moi, area)

  Computes the slenderness ratio of a column.
- pure elemental real(real64) function euler_buckling_1 (modulus, ratio, area)

  Computes the critical buckling load of a column by means of Euler's equation.
- pure elemental real(real64) function euler_buckling_2 (modulus, length, moi, area)

  Computes the critical buckling load of a column by means of Euler's equation.
- pure elemental real(real64) function johnson_buckling_1 (modulus, yield, ratio, area)

  Computes the critical buckling load of a column by means of J.B. Johnson's equation.
- pure elemental real(real64) function johnson_buckling_2 (modulus, yield, length, moi, area)

  Computes the critical buckling load of a column by means of J.B. Johnson's equation.
- pure elemental real(real64) function, public buckling_transition (modulus, yield)

  Computes the slenderness ratio transition point above which the Euler equation is more appropriate to estimate the buckling load, but below which the Johnson equation is more valid.
- pure elemental real(real64) function buckling_load_1 (modulus, yield, ratio, area)

  Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_↩ transition routine to determine where the transition between the two equations lies.
- pure elemental real(real64) function buckling_load_2 (modulus, yield, length, moi, area)

  Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_↩ transition routine to determine where the transition between the two equations lies.

3.1.1 Detailed Description

columns

Purpose

This module contains calculations relating to buckling and stability of columns in compression.

References

-

3.1.2 Function/Subroutine Documentation

3.1.2.1 pure elemental real(real64) function columns::buckling_load_1 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *ratio*, real(real64), intent(in) *area*) [private]

Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_transition routine to determine where the transition between the two equations lies.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 214 of file columns.f90.

3.1.2.2 pure elemental real(real64) function columns::buckling_load_2 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_transition routine to determine where the transition between the two equations lies.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the section.

Returns

The critical buckling load.

Definition at line 245 of file columns.f90.

3.1.2.3 pure elemental real(real64) function, public columns::buckling_transition (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*)

Computes the slenderness ratio transition point above which the Euler equation is more appropriate to estimate the buckling load, but below which the Johnson equation is more valid.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.

Definition at line 192 of file columns.f90.

3.1.2.4 pure elemental real(real64) function columns::euler_buckling_1 (real(real64), intent(in) *modulus*, real(real64), intent(in) *ratio*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of Euler's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column was made.
in	*ratio*	The slenderness ratio of the column.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 120 of file columns.f90.

3.1.2.5 pure elemental real(real64) function columns::euler_buckling_2 (real(real64), intent(in) *modulus*, real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of Euler's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column was made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 137 of file columns.f90.

3.1.2.6 pure elemental real(real64) function columns::johnson_buckling_1 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *ratio*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of J.B. Johnson's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*ratio*	The slenderness ratio of the column.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 156 of file columns.f90.

3.1.2.7 pure elemental real(real64) function columns::johnson_buckling_2 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of J.B. Johnson's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 175 of file columns.f90.

3.1.2.8 pure elemental real(real64) function columns::slenderness_ratio_1 (real(real64), intent(in) *length*, real(real64), intent(in) *rad*) [private]

Computes the slenderness ratio of a column.

Parameters

in	*length*	The effective length of the column.
in	*rad*	The radius of gyration of the section.

Returns

The slenderness ratio.

Definition at line 91 of file columns.f90.

3.1.2.9 pure elemental real(real64) function columns::slenderness_ratio_2 (real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the slenderness ratio of a column.

Parameters

in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the section.

Returns

The slenderness ratio.

Definition at line 104 of file columns.f90.

3.2 constants Module Reference

constants

Variables

- real(real64), parameter pi = 3.14159265358979323846264338327950d0
  The parameter pi.

3.2.1 Detailed Description

constants

Purpose

This module provides constants commonly used in engineering calculations.

3.3 sections Module Reference

sections

Functions/Subroutines

- pure elemental real(real64) function radius_of_gyration (moi, area)
  Computes the radius of gyration of a section.
- pure real(real64) function, dimension(4) i_beam_section (webt, webh, flanget, flangew)
  Computes the cross-sectional properties of an I-beam section.
- pure real(real64) function, dimension(3) circle_section (diameter)
  Computes the cross-sectional properties of a circular section.
- pure real(real64) function, dimension(3) hollow_circle_section (od, id)
  Computes the cross-sectional properties of a hollow circular section.

3.3.1 Detailed Description

sections

Purpose

This module contains routines used for computing the sectional properties of various shapes.

3.3.2 Function/Subroutine Documentation

3.3.2.1 pure real(real64) function, dimension(3) sections::circle_section (real(real64), intent(in) *diameter*)

Computes the cross-sectional properties of a circular section.

Parameters

`in`	*diameter*	The diameter of the section.

Returns

An array containing the following cross-sectional properties.

- Cross-sectional area.
- Moment of inertia.
- Polar moment of inertia.

The cross-sectional properties are computed as follows.

- Area:
  - $A = \frac{\pi}{4}d^2$
- Moment of Inertia:
  - $I = \frac{\pi}{64}d^4$
- Polar Moment of Inertia:
  - $J = \frac{\pi}{32}d^4$

Definition at line 97 of file sections.f90.

3.3.2.2 pure real(real64) function, dimension(3) sections::hollow_circle_section (real(real64), intent(in) *od*, real(real64), intent(in) *id*)

Computes the cross-sectional properties of a hollow circular section.

Parameters

`in`	*od*	The outer diameter of the section.
`in`	*id*	The inner diameter of the section.

Returns

An array containing the following cross-sectional properties.

- Cross-sectional area.
- Moment of inertia.
- Polar moment of inertia.

The cross-sectional properties are computed as follows.

- Area:
  - $A = \frac{\pi}{4}(d_o^2 - d_i^2)$
- Moment of Inertia:
  - $I = \frac{\pi}{64}(d_o^4 - d_i^4)$
- Polar Moment of Inertia:
  - $J = \frac{\pi}{32}(d_o^4 - d_i^4)$

Definition at line 131 of file sections.f90.

3.3.2.3 pure real(real64) function, dimension(4) sections::i_beam_section (real(real64), intent(in) *webt*, real(real64), intent(in) *webh*, real(real64), intent(in) *flanget*, real(real64), intent(in) *flangew*)

Computes the cross-sectional properties of an I-beam section.

Parameters

in	*webt*	The web section thickness.
in	*webh*	The web section height (between the flanges).
in	*flanget*	The flange thickness.
in	*flangew*	The flange width.

Returns

An array containing the following cross-sectional properties.

- Cross-sectional area.
- Moment of inertia about the x-axis of the section (Stiff Axis).
- Moment of inertia about the y-axis of the section (Soft Axis).
- Polar moment of inertia.

The cross-sectional properties are computed as follows.

- Area:
  - $A = t_w h_w + 2 w_f t_f$
- Moments of Inertia:
  - $I_x = \int y^2 \, dA = \frac{1}{12}(t_w h_w^3 + 2 w_f t_f^3) + \frac{1}{2} w_f t_f (h_w + t_f)^2$
  - $I_y = \int x^2 \, dA = \frac{1}{12}(h_w t_w^3 + 2 t_f w_f^3)$
- Polar Moment of Inertia:
  - $J = \int r^2 \, dA = \int (x^2 + y^2) \, dA = I_x + Iy.$

Definition at line 55 of file sections.f90.

3.3.2.4 pure elemental real(real64) function sections::radius_of_gyration (real(real64), intent(in) *moi*, real(real64), intent(in) *area*)

Computes the radius of gyration of a section.

Parameters

in	*moi*	The moment of inertia of area for the section.
in	*area*	The cross-sectional area of the section.

Returns

The radius of gyration for the section.

The radius of gyration is computed as follows.

$$\rho = \sqrt{\frac{I}{A}}$$

Definition at line 26 of file sections.f90.

3.4 strain Module Reference

strain

Data Types

- interface wheatstone_bridge
  Computes the output of a wheatstone bridge.

Functions/Subroutines

- pure elemental real(real64) function wheatstone_bridge_1 (fg, strain1, strain2, strain3, strain4)
  Computes the output of a wheatstone bridge.
- pure real(real64) function wheatstone_bridge_2 (fg, r, strain1, strain2, strain3, strain4)
  Computes the output of a wheatstone bridge.
- pure elemental real(real64) function strain_transform_x (ex, ey, gxy, theta)
  Applies a rotation transformation to the x-direction strain.
- pure elemental real(real64) function strain_transform_y (ex, ey, gxy, theta)
  Applies a rotation transformation to the y-direction strain.
- pure elemental real(real64) function strain_transfom_xy (ex, ey, gxy, theta)
  Applies a rotation transformation to the x-y shear strain.

3.4.1 Detailed Description

strain

Purpose

Provides routines for performing strain related calculations.

3.4.2 Function/Subroutine Documentation

3.4.2.1 pure elemental real(real64) function strain::strain_transfom_xy (real(real64), intent(in) *ex*, real(real64), intent(in) *ey*, real(real64), intent(in) *gxy*, real(real64), intent(in) *theta*) [private]

Applies a rotation transformation to the x-y shear strain.

Parameters

in	*ex*	The x-direction strain.
in	*ey*	The y-direction strain.
in	*gxy*	The x-y shear strain.
in	*theta*	The rotation angle, in radians.

Returns

The transformed strain.

Definition

$$\gamma_{x'y'} = 2(\epsilon_x - \epsilon_y)\sin\theta\cos\theta + \gamma_{xy}(\cos^2\theta - \sin^2\theta)$$

Definition at line 172 of file strain.f90.

3.4.2.2 pure elemental real(real64) function strain::strain_transform_x (real(real64), intent(in) *ex*, real(real64), intent(in) *ey*, real(real64), intent(in) *gxy*, real(real64), intent(in) *theta*) [private]

Applies a rotation transformation to the x-direction strain.

Parameters

in	*ex*	The x-direction strain.
in	*ey*	The y-direction strain.
in	*gxy*	The x-y shear strain.
in	*theta*	The rotation angle, in radians.

Returns

The transformed strain.

Definition

$$\epsilon_{x'} = \epsilon_x\cos^2\theta + \epsilon_y\sin^2\theta + \gamma_{xy}\sin\theta\cos\theta$$

Definition at line 132 of file strain.f90.

3.4.2.3 pure elemental real(real64) function strain::strain_transform_y (real(real64), intent(in) *ex*, real(real64), intent(in) *ey*, real(real64), intent(in) *gxy*, real(real64), intent(in) *theta*) [private]

Applies a rotation transformation to the y-direction strain.

Parameters

in	*ex*	The x-direction strain.
in	*ey*	The y-direction strain.
in	*gxy*	The x-y shear strain.
in	*theta*	The rotation angle, in radians.

Returns

The transformed strain.

Definition

$$\epsilon_{y'} = \epsilon_x\sin^2\theta + \epsilon_y\cos^2\theta - \gamma_{xy}\sin\theta\cos\theta$$

Definition at line 152 of file strain.f90.

3.4.2.4 pure elemental real(real64) function strain::wheatstone_bridge_1 (real(real64), intent(in) *fg*, real(real64), intent(in) *strain1*, real(real64), intent(in) *strain2*, real(real64), intent(in) *strain3*, real(real64), intent(in) *strain4*) [private]

Computes the output of a wheatstone bridge.

Parameters

in	*fg*	The gage factor of each strain gage. Each gage is assigned the same gage factor.
in	*strain1*	The strain in leg 1 of the bridge.
in	*strain2*	The strain in leg 2 of the bridge.
in	*strain3*	The strain in leg 3 of the bridge.
in	*strain4*	The strain in leg 4 of the bridge.

Returns

The output of the bridge, in units of mV/V.

Remarks

The construction of the wheatstone bridge is as follows.

```
            V(+)
             /\
   Leg 1  /   \ Leg 4
Vout(+) _/     \_ Vout(-)
          \     /
   Leg 2  \   / Leg 3
             \/
            V(-)
```

The output of the bridge assuming each gage has the same resistance and the same gage factor is as follows.

$$\frac{V_{out}}{V} = \frac{f_g(f_g(\epsilon_1\epsilon_3 - \epsilon_2\epsilon_4) - (\epsilon_4 - \epsilon_3 + \epsilon_2 - \epsilon_1))}{4 + f_g(f_g(\epsilon_3(\epsilon_2+\epsilon_1) + \epsilon_4(\epsilon_2+\epsilon_1)) + 2(\epsilon_1+\epsilon_2+\epsilon_3+\epsilon_4))}$$

Definition at line 56 of file strain.f90.

3.4.2.5 pure real(real64) function strain::wheatstone_bridge_2 (real(real64), dimension(4), intent(in) *fg*, real(real64), dimension(4), intent(in) *r*, real(real64), intent(in) *strain1*, real(real64), intent(in) *strain2*, real(real64), intent(in) *strain3*, real(real64), intent(in) *strain4*) [private]

Computes the output of a wheatstone bridge.

Parameters

in	*fg*	A 4-element array containing the gage factor for each strain gage (gage 1, gage 2, gage 3, gage 4).
in	*r*	A 4-element array containing the resistance for each strain gage (gage 1, gage 2, gage 3, gage 4).
in	*strain1*	The strain in leg 1 of the bridge.
in	*strain2*	The strain in leg 2 of the bridge.
in	*strain3*	The strain in leg 3 of the bridge.
in	*strain4*	The strain in leg 4 of the bridge.

Returns

The output of the bridge, in units of mV/V.

Remarks

The construction of the wheatstone bridge is as follows.

```
            V(+)
            /\
   Leg 1   /  \ Leg 4
Vout(+) _/     \_ Vout(-)
         \     /
   Leg 2  \   / Leg 3
            \/
            V(-)
```

The output of the bridge is defined as follows.

$$\frac{V_{out}}{V} = \frac{R_3+\Delta R_3}{R_3+\Delta R_3+R_4+\Delta R_4} - \frac{R_2+\Delta R_2}{R_2+\Delta R_2+R_1+\Delta R_1}$$

Where: $\Delta R_i = f_{g,i} R_i \epsilon_i, i = 1..4$.

Definition at line 103 of file strain.f90.

4 Data Type Documentation

4.1 columns::buckling_load Interface Reference

Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column.

Private Member Functions

- pure elemental real(real64) function buckling_load_1 (modulus, yield, ratio, area)

  Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_↩ transition routine to determine where the transition between the two equations lies.
- pure elemental real(real64) function buckling_load_2 (modulus, yield, length, moi, area)

  Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_↩ transition routine to determine where the transition between the two equations lies.

4.1.1 Detailed Description

Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column.

Definition at line 78 of file columns.f90.

4.1.2 Member Function/Subroutine Documentation

4.1.2.1 pure elemental real(real64) function columns::buckling_load::buckling_load_1 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *ratio*, real(real64), intent(in) *area*) [private]

Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_transition routine to determine where the transition between the two equations lies.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 214 of file columns.f90.

4.1.2.2 pure elemental real(real64) function columns::buckling_load::buckling_load_2 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the buckling load of a column under compression. The routine determines whether to use the J.B. Johnson equation or the Euler equation based upon the slenderness ratio of the column. The routine uses the buckling_transition routine to determine where the transition between the two equations lies.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the section.

Returns

The critical buckling load.

Definition at line 245 of file columns.f90.

The documentation for this interface was generated from the following file:

- /home/jason/Documents/Code/mecheng/src/columns.f90

4.2 columns::euler_buckling Interface Reference

Computes the critical buckling load of a column by means of Euler's equation.

Private Member Functions

- pure elemental real(real64) function euler_buckling_1 (modulus, ratio, area)

  Computes the critical buckling load of a column by means of Euler's equation.
- pure elemental real(real64) function euler_buckling_2 (modulus, length, moi, area)

  Computes the critical buckling load of a column by means of Euler's equation.

4.2.1 Detailed Description

Computes the critical buckling load of a column by means of Euler's equation.

Remarks

The Euler buckling equation is as follows: $P_{cr} = \frac{\pi^2 EA}{R_s^2}$; where $R_s = \frac{L_e}{\rho}$, and $\rho = \sqrt{\frac{I}{A}}$.

- $P_{cr} =$ The critical buckling load.
- $E =$ The modulus of elasticity of the material.
- $L_e =$ The effective length of the column.
- $I =$ The moment of inertia (area) of the column (softest axis).
- $A =$ The cross-sectional area of the column.

Definition at line 49 of file columns.f90.

4.2.2 Member Function/Subroutine Documentation

4.2.2.1 pure elemental real(real64) function columns::euler_buckling::euler_buckling_1 (real(real64), intent(in) *modulus*, real(real64), intent(in) *ratio*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of Euler's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column was made.
in	*ratio*	The slenderness ratio of the column.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 120 of file columns.f90.

4.2.2.2 pure elemental real(real64) function columns::euler_buckling::euler_buckling_2 (real(real64), intent(in) *modulus*, real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of Euler's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column was made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 137 of file columns.f90.

The documentation for this interface was generated from the following file:

- /home/jason/Documents/Code/mecheng/src/columns.f90

4.3 columns::johnson_buckling Interface Reference

Computes the critical buckling load of a column by means of J.B. Johnson's equation.

Private Member Functions

- pure elemental real(real64) function johnson_buckling_1 (modulus, yield, ratio, area)
  Computes the critical buckling load of a column by means of J.B. Johnson's equation.
- pure elemental real(real64) function johnson_buckling_2 (modulus, yield, length, moi, area)
  Computes the critical buckling load of a column by means of J.B. Johnson's equation.

4.3.1 Detailed Description

Computes the critical buckling load of a column by means of J.B. Johnson's equation.

Remarks

The J.B. Johnson buckling equation is as follows: $P_{cr} = A(S_y - (\frac{S_y R_s}{2\pi})^2(\frac{1}{E}))$; where $R_s = \frac{L_e}{\rho}$, and $\rho = \sqrt{\frac{I}{A}}$.

- $P_{cr} =$ The critical buckling load.
- $S_y =$ The yield strength of the material.
- $E =$ The modulus of elasticity of the material.
- $L_e =$ The effective length of the column.
- $I =$ The moment of inertia (area) of the column (softest axis).
- $A =$ The cross-sectional area of the column.

Definition at line 69 of file columns.f90.

4.3.2 Member Function/Subroutine Documentation

4.3.2.1 pure elemental real(real64) function columns::johnson_buckling::johnson_buckling_1 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *ratio*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of J.B. Johnson's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*ratio*	The slenderness ratio of the column.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 156 of file columns.f90.

4.3.2.2 pure elemental real(real64) function columns::johnson_buckling::johnson_buckling_2 (real(real64), intent(in) *modulus*, real(real64), intent(in) *yield*, real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the critical buckling load of a column by means of J.B. Johnson's equation.

Parameters

in	*modulus*	The modulus of elasticity of the material from which the column is made.
in	*yield*	The yield strength of the material from which the column is made.
in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the column.

Returns

The critical buckling load.

Definition at line 175 of file columns.f90.

The documentation for this interface was generated from the following file:

- /home/jason/Documents/Code/mecheng/src/columns.f90

4.4 columns::slenderness_ratio Interface Reference

Computes the slenderness ratio of a column.

Private Member Functions

- pure elemental real(real64) function slenderness_ratio_1 (length, rad)

  Computes the slenderness ratio of a column.
- pure elemental real(real64) function slenderness_ratio_2 (length, moi, area)

  Computes the slenderness ratio of a column.

4.4.1 Detailed Description

Computes the slenderness ratio of a column.

Remarks

The slenderness ratio is defined as the ratio between the effective length of the column (L_e), and the radius of gyration of the column (ρ): $R_s = \frac{L_e}{\rho}$.

Definition at line 30 of file columns.f90.

4.4.2 Member Function/Subroutine Documentation

4.4.2.1 pure elemental real(real64) function columns::slenderness_ratio::slenderness_ratio_1 (real(real64), intent(in) *length*, real(real64), intent(in) *rad*) [private]

Computes the slenderness ratio of a column.

Parameters

in	*length*	The effective length of the column.
in	*rad*	The radius of gyration of the section.

Returns

The slenderness ratio.

Definition at line 91 of file columns.f90.

4.4.2.2 pure elemental real(real64) function columns::slenderness_ratio::slenderness_ratio_2 (real(real64), intent(in) *length*, real(real64), intent(in) *moi*, real(real64), intent(in) *area*) [private]

Computes the slenderness ratio of a column.

Parameters

in	*length*	The effective length of the column.
in	*moi*	The moment of inertia (area) for the section.
in	*area*	The cross-sectional area of the section.

Returns

The slenderness ratio.

Definition at line 104 of file columns.f90.

The documentation for this interface was generated from the following file:

- /home/jason/Documents/Code/mecheng/src/columns.f90

4.5 strain::wheatstone_bridge Interface Reference

Computes the output of a wheatstone bridge.

Private Member Functions

- pure elemental real(real64) function wheatstone_bridge_1 (fg, strain1, strain2, strain3, strain4)
  Computes the output of a wheatstone bridge.
- pure real(real64) function wheatstone_bridge_2 (fg, r, strain1, strain2, strain3, strain4)
  Computes the output of a wheatstone bridge.

4.5.1 Detailed Description

Computes the output of a wheatstone bridge.

Definition at line 15 of file strain.f90.

4.5.2 Member Function/Subroutine Documentation

4.5.2.1 pure elemental real(real64) function strain::wheatstone_bridge::wheatstone_bridge_1 (real(real64), intent(in) *fg*, real(real64), intent(in) *strain1*, real(real64), intent(in) *strain2*, real(real64), intent(in) *strain3*, real(real64), intent(in) *strain4*) [private]

Computes the output of a wheatstone bridge.

Parameters

in	*fg*	The gage factor of each strain gage. Each gage is assigned the same gage factor.
in	*strain1*	The strain in leg 1 of the bridge.
in	*strain2*	The strain in leg 2 of the bridge.
in	*strain3*	The strain in leg 3 of the bridge.
in	*strain4*	The strain in leg 4 of the bridge.

Returns

The output of the bridge, in units of mV/V.

Remarks

The construction of the wheatstone bridge is as follows.

```
            V(+)
            /\
   Leg 1   /  \ Leg 4
Vout(+)  _/    \_ Vout(-)
          \    /
   Leg 2   \  / Leg 3
            \/
            V(-)
```

The output of the bridge assuming each gage has the same resistance and the same gage factor is as follows.

$$\frac{V_{out}}{V} = \frac{f_g(f_g(\epsilon_1 \epsilon_3 - \epsilon_2 \epsilon_4) - (\epsilon_4 - \epsilon_3 + \epsilon_2 - \epsilon_1))}{4 + f_g(f_g(\epsilon_3(\epsilon_2 + \epsilon_1) + \epsilon_4(\epsilon_2 + \epsilon_1)) + 2(\epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4))}$$

Definition at line 56 of file strain.f90.

4.5.2.2 pure real(real64) function strain::wheatstone_bridge::wheatstone_bridge_2 (real(real64), dimension(4), intent(in) *fg*, real(real64), dimension(4), intent(in) *r*, real(real64), intent(in) *strain1*, real(real64), intent(in) *strain2*, real(real64), intent(in) *strain3*, real(real64), intent(in) *strain4*) `[private]`

Computes the output of a wheatstone bridge.

Parameters

in	*fg*	A 4-element array containing the gage factor for each strain gage (gage 1, gage 2, gage 3, gage 4).
in	*r*	A 4-element array containing the resistance for each strain gage (gage 1, gage 2, gage 3, gage 4).
in	*strain1*	The strain in leg 1 of the bridge.
in	*strain2*	The strain in leg 2 of the bridge.
in	*strain3*	The strain in leg 3 of the bridge.
in	*strain4*	The strain in leg 4 of the bridge.

Returns

The output of the bridge, in units of mV/V.

Remarks

The construction of the wheatstone bridge is as follows.

```
           V(+)
           /\
    Leg 1  /  \ Leg 4
Vout(+) _/    \_ Vout(-)
         \    /
    Leg 2  \  / Leg 3
           \/
           V(-)
```

The output of the bridge is defined as follows.

$$\frac{V_{out}}{V} = \frac{R_3 + \Delta R_3}{R_3 + \Delta R_3 + R_4 + \Delta R_4} - \frac{R_2 + \Delta R_2}{R_2 + \Delta R_2 + R_1 + \Delta R_1}$$

Where: $\Delta R_i = f_{g,i} R_i \epsilon_i, i = 1..4$.

Definition at line 103 of file strain.f90.

The documentation for this interface was generated from the following file:

- /home/jason/Documents/Code/mecheng/src/strain.f90

Index