Package 'iemisc'

June 5, 2024

Title Irucka Embry's Miscellaneous Functions

Version 1.0.5

Maintainer Irucka Embry <iembry@ecoccs.com>

Depends R (>= 3.3.0)

Imports zoo, pracma, iemiscdata, gsubfn (>= 0.7), fpCompare, units (>= 0.7-0), stringi, assertthat, rivr (>= 1.2-2), checkmate, chem.databases, methods, ramify, foreach, stats, data.table (>= 1.10.2), measurements, roperators, berryFunctions, round, USA.state.boundaries (>= 1.0.1), sf, ggplot2, ggpubr, matlab, sjmisc, lubridate, anytime, mgsub, geosphere, matlab2r, signal, utils, qdapRegex

Suggests install.load, knitr, import, fractional, fracture, MASS, rmarkdown, tinytest, maps, spelling, sampler, callr, rando, geometry, linguisticsdown, aiRthermo, hydraulics, ie2misc, formatR, pander, printr, tibble, lintr, opencpu

Description A collection of Irucka Embry's miscellaneous functions (Engineering Economics, Civil & Environmental/Water Resources Engineering, Construction Measurements, GNU Octave compatible functions, Python compatible function, Trigonometric functions in degrees and function in radians, Geometry, Statistics, Mortality Calculators, Quick Search, etc.).

URL https://gitlab.com/iembry/iemisc

BugReports https://gitlab.com/iembry/iemisc/-/issues

License GPL (>= 3) | file LICENSE

Language en-US

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.3.1

NeedsCompilation no

Author Irucka Embry [aut, cre],
Felix Andrews [aut, ctb] (zoo code),
Hans Werner Borchers [aut, ctb] (pracma code),

Samit Basu [aut, ctb] (FreeMat code),
David Bateman [aut, ctb] (GNU Octave code),
Rik Wehbring [aut, ctb] (GNU Octave code),
Didier Clamond [aut, ctb] (MATLAB code),
Berry Boessenkool [aut, ctb] (checkLL and degree functions from OSMscale),
Tyler Rinker [aut, ctb] (lookup and lookup_helper functions from qdapTools),
Colin Caprani [aut, ctb] (secprop MATLAB function),
Teodor Ciuraru [aut, ctb] (stackoverflow R code),
Dylan Russell [aut, ctb] (stackoverflow R code),
John Wallace [aut, ctb] (stackoverflow R code),
kaijagahm [aut, ctb] (R bloggers R code),
Colin B. Macdonald [aut, ctb] (OctSymPy code),
John D Page [aut, ctb] (JavaScript code),
Josh O'Brien [aut, ctb] (stackoverflow R code),
R. van Twisk [aut, ctb] (LibreCAD code)

Repository CRAN

Date/Publication 2024-06-05 21:40:18 UTC

Contents

`acosd` *Inverse cosine (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse cosine for each element of `x` in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
acosd(x)
```

Arguments

`x`	A numeric vector containing values in degrees

Value

The inverse cosine of each element of `x` in degrees.

Note

Note: If you have a radian (rad) angle value, use `acos` instead.

Author(s)

David Bateman (GNU Octave acosd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. `https://docs.octave.org/octave.pdf`. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave acosd
acosd (seq(0, 1, by = 0.1))
```

acotd — *Inverse cotangent (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse cotangent for each element of x in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
acotd(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The inverse cotangent of each element of x in degrees.

Note

Note: If you have a radian (rad) angle value, use atan instead.

Author(s)

David Bateman (GNU Octave acotd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave acotd
acotd (seq(10, 90, by = 10))
```

---

acscd *Inverse cosecant (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse cosecant for each element of `x` in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
acscd(x)
```

Arguments

`x`	A numeric vector containing values in degrees

Value

The inverse cosecant of each element of `x` in degrees.

Note

Note: If you have a radian (rad) angle value, use `acsc` instead.

Author(s)

David Bateman (GNU Octave acscd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave acscd
acscd (seq(0, 90, by = 10))
```

AgivenF *Annual value given Future value (Engineering Economics)*

Description

Compute A given F

Usage

```
AgivenF(
  F,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)

AF(
  F,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

F	numeric vector that contains the future value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as a percent
frequency	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

A is expressed as

$$A = F \left[\frac{i}{\left(1 + i\right)^n - 1} \right]$$

A the "uniform series amount (occurs at the end of each interest period)"

F the "future equivalent"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

AgivenF numeric vector that contains the annual value(s) rounded to 2 decimal places

AF data.frame of both n (0 to n) and the resulting annual values rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 135-136, 142, 164.

Examples

```
library(iemisc)

# Example for equation 4-12 from the Reference text (page 135-136)
AgivenF(309*10^6, 60, 0.5, "month")

# the interest rate is 0.5\% per month and n is 60 months
# "$4.4187 million per month" is the answer

AF(309*10^6, 60, 0.5, "annual")
# the interest rate is 0.5\% per month and n is 60 months
```

`AgivenFcont` *Annual value given Future value [continuous] (Engineering Economics)*

Description

Compute A given F with interest compounded continuously

Usage

```
AgivenFcont(F, n, r)
```

Arguments

`F`	numeric vector that contains the future value(s)
`n`	numeric vector that contains the period value(s)
`r`	numeric vector that contains the continuously compounded nominal annual interest rate(s) as a percent

Details

A is expressed as

$$A = F \left[\frac{e^r - 1}{e^{rn} - 1}\right]$$

A the "annual equivalent amount (occurs at the end of each year)"

F the "future equivalent"

r the "nominal annual interest rate, compounded continuously"

n the "number of periods (years)"

Value

AgivenFcont numeric vector that contains the annual value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 169.

Examples

```
library(iemisc)

AgivenFcont(300, 2, 11) # 11\% interest
```

AgivenG *Annual value given Gradient value (Engineering Economics)*

Description

Compute A given G

Usage

```
AgivenG(
  G,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

`G`	numeric vector that contains the gradient value(s)
`n`	numeric vector that contains the period value(s)
`i`	numeric vector that contains the interest rate(s) as a percent
`frequency`	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

$$A = G\left[\frac{1}{i} - \frac{n}{(1+i)^n - 1}\right]$$

A the "uniform series amount (occurs at the end of each interest period)"

G the "uniform gradient amount"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

AgivenG numeric vector that contains the annual value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 142, 150, 152-154, 164, 166-167.

Examples

```
library(iemisc)

# Example 4-20 from the Reference text (pages 153-154)
  AgivenG(1000, 4, 15, "annual") # the interest rate is 15\%


# Example 4-31 from the Reference text (pages 166-167)
 AgivenG(1000, 4, 20, "semiannual") # the nominal interest rate is 20\% compounded semiannually
```

AgivenP — *Annual value given Present value (Engineering Economics)*

Description

Compute A given P

Usage

```
AgivenP(
  P,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)

AP(
  P,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

P	numeric vector that contains the present value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as a percent
frequency	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

A is expressed as

$$A = P\left[\frac{i\left(1+i\right)^{n}}{\left(1+i\right)^{n}-1}\right]$$

A the "uniform series amount (occurs at the end of each interest period)"

P the "present equivalent"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

AgivenP numeric vector that contains the annual value(s) rounded to 2 decimal places

AP data.frame of both n (0 to n) and the resulting annual values rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 136, 142, 164, 166.

Examples

```
library(iemisc)

# Example for equation 4-14 from the Reference text (page 136)
AgivenP(17000, 4, 1, "annual")
# the interest rate is 1\% per month and n is 4 months

AP(17000, 4, 1, "annual")
# the interest rate is 1\% per month and n is 4 months


# Example 4-30 from the Reference text (page 166)
AgivenP(10000, 5, 12, "month")
# the interest rate is 12% compounded monthly for 5 years

AP(10000, 5, 12, "month")
# the interest rate is 12% compounded monthly for 5 years
```

`AgivenPcont` *Annual value given Present value [continuous] (Engineering Economics)*

Description

Compute A given P with interest compounded continuously

Usage

```
AgivenPcont(P, n, r)
```

Arguments

`P`	numeric vector that contains the present value(s)
`n`	numeric vector that contains the period value(s)
`r`	numeric vector that contains the continuously compounded nominal annual interest rate(s) as a percent

Details

A is expressed as

$$A = P\left[\frac{e^{rn}\left(e^{r}-1\right)}{e^{rn}-1}\right]$$

A the "annual equivalent amount (occurs at the end of each year)"

P the "present equivalent"

r the "nominal annual interest rate, compounded continuously"

n the "number of periods (years)"

Value

AgivenPcont numeric vector that contains the annual value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 169-170.

Examples

```
library(iemisc)

# Example for equation 4-34 from the Reference text (page 170)
AgivenPcont(1000, 10, 20) # 20\% interest
```

---

air_stripper *Design of Packed Column Air Strippers*

Description

Calculates key parameters needed in the design of a packed column air stripper according to the U.S. Army Corps of Engineers Design Guide No. 1110-1-3: Air Stripping Engineering and Design (Design Guidelines). Please refer to the Design Guidelines for the governing equations and background information.

'Air stripping is the transferring of volatile components of a liquid into an air stream. It is an environmental engineering technology used for the purification of groundwaters and wastewaters containing volatile compounds.' (Reference: Wikipedia)

Usage

```
air_stripper(
  Temp,
  pTe,
  contam1,
  Cai,
  Cae,
  contam2,
  cas = NULL,
  Ha,
  Q,
  loading,
  ns,
  DL,
  DG,
  R,
  P_atm = NULL,
  dP = NULL,
  at = NULL,
  Sc = NULL,
  cf = NULL,
  Temp_unit = c("SI", "Eng", "Absolute"),
  dP_unit = c("inch", "mm"),
  at_unit = c("ft^2/ft^3", "m^2/m^3"),
```

```
  Sc_unit = c("kg/s^2", "slug/s^2"),
  contaminants_table = c(0, 1),
  removal_requirements_table = c(0, 1),
  critical_contaminant_table = c(0, 1)
)
```

Arguments

Temp	numeric vector that contains the minimum Temperature (degrees Celsius, degrees Fahrenheit, or Kelvin)
pTe	numeric vector that contains the total pressure of gas (air) effluent (atm)
contam1	character vector that contains the name of each contaminant to be removed (may include "Total VOCs"). See the example.
Cai	numeric vector that contains the concentration of each contaminant in liquid (water) influent (ug/L)
Cae	numeric vector that contains the concentration of each contaminant in liquid (water) effluent (ug/L)
contam2	character vector that contains the name of each contaminant (will not include "Total VOCs"). See the example.
cas	character vector that contains the CAS Number of each contaminant, if known, otherwise it can be accessed internally
Ha	numeric vector that contains the Ha (Henry's Law coefficient) at the minimum Temperature *T* for each contaminant (atm/mole/mole)
Q	numeric vector that contains the sustained pumping rate (gallons per minute or gpm)
loading	numeric vector that contains the stripper surface loading (gpm/ft^2)
ns	numeric vector that contains the number of strippers
DL	numeric vector that contains the liquid diffusivity of each contaminant at the minimum Temperature *T* in the same order as contam2 (m^2/s)
DG	numeric vector that contains the gas diffusivity of each contaminant in air at the minimum Temperature *T* and *pTe* in the same order as contam2 (m^2/s)
R	numeric vector that contains the stripping factor (R = 2.5 if air pollution control is required or R = 4.5 if it isn't, but in a range of 2 - 5)
P_atm	numeric vector that contains the atmospheric pressure (atm). The default is 1 atm.
dP	numeric vector that contains the nominal diameters for the packing material. The Design Guidelines use Jaeger Tripacks 2-in. (50.8 mm) plastic media for the packing material.
at	numeric vector that contains the total surface area for the packing material (the default value is 157 m^2/m^3)
Sc	numeric vector that contains the critical surface tension for the packing material (the default value is 0.033 kg/s^2)
cf	numeric vector that contains the packing factor for the packing material (the default value is 15/ft)

`Temp_unit` character vector that contains the possible units for the water temperature [options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), or `Absolute` for Absolute Units]

`dP_unit` character vector that contains the possible units for the nominal diameters for the packing material (inch or mm)

`at_unit` character vector that contains the possible units for the total surface area for the packing material (ft^2/ft^3 or m^2/m^3)

`Sc_unit` character vector that contains the possible units for the critical surface tension for the packing material (kg/s^2 or slug/s^2)

`contaminants_table`
integer vector that contains 0, 1 only. 0 represents do not print the Contaminants Table and 1 is for printing the Contaminants Table.

`removal_requirements_table`
integer vector that contains 0, 1 only. 0 represents do not print the Removal Requirements Table and 1 is for printing the Removal Requirements Table.

`critical_contaminant_table`
integer vector that contains 0, 1 only. 0 represents do not print the Critical Contaminant Table and 1 is for printing the Critical Contaminant Table.

Value

the name of the critical contaminant, molar liquid (water) flow per unit of stripper cross-sectional area (kg mole/m^2 s), molar gas (air) flow per unit of stripper cross-sectional area (kg mole/m^2 s), height of transfer unit (HTU) [m and ft], number of transfer units (NTU), packing depth (m and feet), and the air to water ratio as a `data.table`. If contaminants_table = 1, provide the Contaminants Table. If removal_requirements_table = 1, provide the Removal Requirements Table. If critical_contaminant_table = 1, provide the Critical Contaminant Table.

Note

Please Note: Use these results as preliminary estimates only.

Please Note: This is not meant for any actual designs.

Please Note: The calculations assume dry air rather than humid air.

Please refer to the iemisc: Air Stripping By Packed Column Examples vignette for examples

Author(s)

Irucka Embry

References

1. Accu Dyne Test: Diversified Enterprises. Critical Surface Tension and Contact Angle with Water for Various Polymers, `https://www.accudynetest.com/polytable_03.html`.

2. Design Guide No. 1110-1-3: Air Stripping Engineering and Design Appendix D: Example Air Stripping By Packed Column, Department Of The Army U.S. Army Corps of Engineers, 31 October 2001, pages D-1 - D-18, https://web.archive.org/web/20240217153739/https://www.publications.usace.army.mil/Portals/76/Publications/EngineerDesignGuides/DG_1110-1-3.pdf?ver=2013-08-16-101222-003. Retrieved thanks to the Internet Archive: Wayback Machine
3. Edgar L Andreas, Design Guide No. 1110-1-3: Handbook of Physical Constants and Functions for Use in Atmospheric Boundary Layer Studies, Department Of The Army U.S. Army Corps of Engineers, October 2005, pages D-1 - D-18, https://apps.dtic.mil/sti/pdfs/ADA440352.pdf.
4. EnggCyclopedia. Tutorial: air density calculation, 3 January 2022, https://enggcyclopedia.com/2019/04/air-density-calculation/.
5. Harlan H. Bengtson, PhD, P.E. Continuing Education and Development, Inc., Calculation of Gas Density and Viscosity Course No: H02-008, https://www.scribd.com/document/452763833/Calculation-of-Gas-Density-and-Viscosity-pdf.
6. PCA Series Packed Column Air Strippers, H2K Technologies, Inc., 2011, page 2, http://www.h2ktech.com/pdf_downloads/PCA_Packed_Column_Air_Strippers.pdf.
7. Peter J. Mohr, David B. Newell, and Barry N. Taylor. Continuing Education and Development, Inc., CODATA recommended values of the fundamental physical constants: 2014, *Reviews Of Modern Physics*, Volume 88, July-September 2016, https://web.archive.org/web/20230608140030/https://physics.nist.gov/cuu/pdf/CODATA_JPCRD2016.pdf. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.
8. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://web.archive.org/web/20230427133623/https://physics.nist.gov/cgi-bin/cuu/Value?gn. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.
9. Wikimedia Foundation, Inc. Wikipedia, 27 March 2022, "Air stripping", https://en.wikipedia.org/wiki/Air_stripping.

`approxerror` *Approximate error*

Description

This function computes the "approximate estimate of the error" ("percent relative error").

Usage

```
approxerror(pres, prev)
```

Arguments

`pres`	numeric vector that contains the "present approximation" value(s)
`prev`	numeric vector that contains the "previous approximation" value(s)

Details

Approximate error is expressed as

$$\varepsilon_a = \frac{present\ approximation - previous\ approximation}{present\ approximation} \cdot 100$$

ε_a the "approximate estimate of the error"

present approximation the "present approximation"

previous approximation the "previous approximation"

Value

approximate error, as a percent (%), as a numeric `vector`.

Author(s)

Irucka Embry

References

Steven C. Chapra, *Applied Numerical Methods with MATLAB for Engineers and Scientists*, Second Edition, Boston, Massachusetts: McGraw-Hill, 2008, page 82-84.

See Also

`sgm` for geometric mean, `shm` for harmonic mean, `cv` for coefficient of variation (CV), `rms` for root-mean-square (RMS), `relerror` for relative error, and `ranges` for sample range.

Examples

```
library(iemisc)

# Example 4.1 from the Reference text (page 84)

approxerror(1.5, 1) # answer as a percent (\%)
```

asecd *Inverse secant (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse secant for each element of x in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
asecd(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The inverse secant of each element of x in degrees.

Note

Note: If you have a radian (rad) angle value, use asec instead.

Author(s)

David Bateman (GNU Octave asecd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave asecd
asecd (seq(0, 90, by = 10))
```

asind *Inverse sine (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse sine for each element of x in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
asind(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The inverse sine of each element of x in degrees.

Note

Note: If you have a radian (rad) angle value, use asin instead.

Author(s)

David Bateman (GNU Octave asind), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave asind
asind(seq(0, 1, by = 0.1))
```

atan2d *"Two-argument arc-tangent" (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of the "two-argument arc-tangent" for each element of (y, x) in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
atan2d(y, x)
```

Arguments

y	A numeric vector containing values in degrees
x	A numeric vector containing values in degrees

Value

The "two-argument arc-tangent" of each element of (y, x) in degrees. Note: "The arc-tangent of two arguments atan2(y, x) returns the angle between the x-axis and the vector from the origin to (x, y), i.e., for positive arguments atan2(y, x) == atan(y/x)." Source: `Trig` (base).

Note

Note: If you have a radian (rad) angle value, use `atan2` instead.

Author(s)

Rik Wehbring (GNU Octave atan2d), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave atan2d
atan2d (a <- seq(-1, 1, by = 0.1), b <- seq(1, -1, by = -0.1))
```

`atand` *Inverse tangent (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse tangent for each element of `x` in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
atand(x)
```

Arguments

`x`	A numeric vector containing values in degrees

Value

The inverse tangent of each element of `x` in degrees.

Note

Note: If you have a radian (rad) angle value, use `atan` instead.

Author(s)

David Bateman (GNU Octave atand), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave atand
atand (seq(0, 90, by = 10))
```

benefitcost *Benefit-Cost Ratio (Engineering Economics)*

Description

Compute the benefit-cost ratio between two alternatives

Usage

```
benefitcost(
  ic1,
  n1,
  ac1,
  ab1,
  i1,
  salvage1,
  ic2,
  n2,
  ac2,
  ab2,
  i2,
  salvage2,
  option1,
  option2,
  table = c("ptable", "rtable", "both")
)
```

Arguments

ic1	numeric vector that contains the initial cost for option 1
n1	numeric vector that contains the useful life (years) for option 1
ac1	numeric vector that contains the annual cost [operations & maintenance (O&M)] for option 1
ab1	numeric vector that contains the annual benefits for option 1
i1	numeric vector that contains the effective interest rate per period as a percent for option 1
salvage1	numeric vector that contains the salvage value for option 1
ic2	numeric vector that contains the initial cost for option 2
n2	numeric vector that contains the useful life (years) for option 2
ac2	numeric vector that contains the annual cost [operations & maintenance (O&M)] for option 2
ab2	numeric vector that contains the annual benefits for option 2
i2	numeric vector that contains the effective interest rate per period as a percent for option 2

`salvage2`	numeric vector that contains the salvage value for option 2
`option1`	character vector that contains the option name for option 1
`option2`	character vector that contains the option name for option 2
`table`	character vector that contains the table output format (ptable, rtable, or both)

Details

Benefit is expressed as

$$Benefit = AB\left[\frac{\left(1+i\right)^n - 1}{i\left(1+i\right)^n}\right]$$

Benefit the present equivalent benefit

AB the annual benefit

i the "effective interest rate" per year

n the number of years

Cost is expressed as

$$Cost = PC + OM\left[\frac{\left(1+i\right)^n - 1}{i\left(1+i\right)^n}\right] - S\left[\frac{1}{\left(1+i\right)^n}\right]$$

Cost the present equivalent cost

PC the present or initial cost

OM the annual operations & maintenance cost

S the salvage value

i the "effective interest rate" per year

n the number of years

Benefit-Cost ratio is expressed as

$$BC = \frac{B_2 - B_1}{C_2 - C_1} \geq 1$$

BC the present equivalent cost

B_1 the benefit for alternative 1

B_2 the benefit for alternative 2

C_1 the cost for alternative 1

C_2 the cost for alternative 2

Value

`data.table` with character vectors with the monetary values having thousands separator in a pretty table (ptable) & message with the best option, data.frame with numeric vectors without the thousands separator in regular table (rtable) & a message with the best option, or both options combined in a list

Author(s)

Irucka Embry

References

1. Michael R. Lindeburg, PE, *EIT Review Manual*, Belmont, California: Professional Publications, Inc., 1996, page 14-2, 14-4.
2. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 133, 142, 442-443, 452-453.

Examples

```
library(iemisc)

# Example from Lindeburg Reference text (page 14-4)
benefitcost(ic1 = 300000, n1 = 10, ac1 = 45000, ab1 = 150000, i1 = 10,
salvage1 = 0, ic2 = 400000, n2 = 10, ac2 = 35000, ab2 = 200000, i2 = 10,
salvage2 = 10000, option1 = "A", option2 = "B", table = "rtable")


# This is useful for saving the results as the named data.frame rtable
rtable <- benefitcost(ic1 = 300000, n1 = 10, ac1 = 45000, ab1 = 150000,
i1 = 10, salvage1 = 0, ic2 = 400000, n2 = 10, ac2 = 35000, ab2 = 200000,
i2 = 10, salvage2 = 10000, option1 = "A", option2 = "B", table = "rtable")

rtable



# This is useful for saving the results as the named data.frame ptable
ptable <- benefitcost(ic1 = 300000, n1 = 10, ac1 = 45000, ab1 = 150000,
i1 = 10, salvage1 = 0, ic2 = 400000, n2 = 10, ac2 = 35000, ab2 = 200000,
i2 = 10, salvage2 = 10000, option1 = "A", option2 = "B", table = "ptable")

ptable



# This is useful for saving the results as the named list of 2 data.frames
# called both
both <- benefitcost(ic1 = 300000, n1 = 10, ac1 = 45000, ab1 = 150000,
i1 = 10, salvage1 = 0, ic2 = 400000, n2 = 10, ac2 = 35000, ab2 = 200000,
i2 = 10, salvage2 = 10000, option1 = "A", option2 = "B", table = "both")

both



# Example 10-8 from the Sullivan Reference text (page 452-453)
project <- benefitcost(ic1 = 750000, n1 = 35, ac1 = 120000, ab1 = 245000,
```

```
i1 = 9, salvage1 = 0, ic2 = 625000, n2 = 25, ac2 = 110000, ab2 = 230000,
i2 = 9, salvage2 = 0, option1 = "Project I", option2 = "Project II",
table = "rtable")

project
```

`colebrook` *Accurately calculate the Colebrook-White equation to obtain the Darcy-Weisbach friction factor*

Description

This function "provides the fast, accurate, and robust computation of the Colebrook-White equation" to determine the "Darcy-Weisbach friction factor F". This method is "more efficient than the solution of the Colebrook equation via the Lambert W-function, or the simple approximations." The solution is accurate to "around machine precision for all R > 3 and for all 0 <= K, i.e. in an interval exceeding all values of physical interest." Reference: Clamond

Usage

```
colebrook(Re, K = NULL)
```

Arguments

`Re`	numeric vector that contains the Reynolds number [dimensionless], which should be >= 2300. Reference: Clamond
`K`	numeric vector that contains the "equivalent sand roughness height sand roughness height (material specific roughness) divided by the hydraulic diameters", if known. If not known, the default value is 0. Reference: Clamond

Details

Colebrook-White equation is expressed as

$$\frac{1}{\sqrt{F}} = -2 * log10 \frac{K}{3.7} + \frac{2.51}{R * \sqrt{F}}$$

F Darcy-Weisbach friction factor

K Equivalent sand roughness height (material specific roughness) divided by the hydraulic diameters

R the Reynolds' number (dimensionless)

Value

F Return a numeric vector containing the Darcy-Weisbach friction factor. Reference: Clamond

Author(s)

Didier Clamond (colebrook MATLAB function), Irucka Embry (colebrook R function)

References

1. Steven C. Chapra, *Applied Numerical Methods with MATLAB for Engineers and Scientists*, Second Edition, Boston, Massachusetts: McGraw-Hill, 2008, pages 157-161.
2. Didier Clamond, "Efficient resolution of the Colebrook equation", *Ind. Eng. Chem. Res.*, 2009, 48 (7), pages 3665-3671 https://arxiv.org/abs/0810.5564 and https://math.univ-cotedazur.fr/~didierc/DidPublis/ICR_2009.pdf

See Also

Re1, Re2, Re3, Re4 for the Reynolds number and f1, f2, f3, f4, f5, f6, f7, and f8 for the Darcy friction factor

Examples

```
install.load::load_package("iemisc", "units")

# Example 1 (Reference: Clamond)
F <- colebrook(c(3e3, 7e5, 1e100), 0.01)

F


# Example 2
# 'Determine f for air flow through a smooth, thin tube. The parameters are
# rho = 1.23 kg/m^3, mu = 1.79 x 10^-5 N * s/m^2, D = 0.005 m, V = 40 m/s
# and epsilon = 0.0015 mm.' Reference: Chapra 158

# Determine R (the Reynolds number) first using the following parameters:

rho <- 1.23 # kg/m^3
V <- 40 # m/s
D <- 0.005 # m
mu <- 1.79 * 10^-5 # N * s/m^2

eps <- 0.0015 # mm
eps <- set_units(eps, "mm")
units(eps) <- make_units(m)

Re <- rho * V * D / mu


K <- drop_units(eps) / D
```

```
# with K
fr1 <- colebrook(Re, K); fr1

# without K
fr2 <- colebrook(Re); fr2


# The solution on Chapra 159 and 160 is 'f = 0.02896781017144' which was
# computed using the Newton-Raphson method, Swamee-Jain approximation
# equation, and MATLAB's fzero function.

# Thus,
fm <- 0.02896781017144

# Compute the relative error between fr[1 and 2] (this function) and fm (Chapra).

relerror(fr1, fm)


relerror(fr2, fm)

# compare the relative error with and without K
```

CompIntCharg *Compound Interest Charged (Engineering Economics)*

Description

Computes the total interest paid at the end of n periods using compound interest

Usage

```
CompIntCharg(
  P,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

P	numeric vector that contains the present value(s)
n	numeric vector that contains the period value(s)

i	numeric vector that contains the interest rate(s) as a percent
frequency	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

Compound Interest Charged is expressed as

$$I = P\left(1 + i\right)^n - P$$

P the "principal amount (lent or borrowed)"

I the "total interest paid"

i the "interest rate per interest period"

n the "number of interest periods"

Value

CompIntCharg numeric vector that contains the total interest paid at the end of n periods rounded to 2 decimal places

Author(s)

Irucka Embry

References

1. *SFPE Handbook of Fire Protection Engineering*. 3rd Edition, DiNenno, P. J.; Drysdale, D.; Beyler, C. L.; Walton, W. D., Editor(s), page 5-94, 2002. Chapter 7; Section 5; NFPA HFPE-02. See https://web.archive.org/web/20180127185316/http://fire.nist.gov/bfrlpubs/build02/PDF/b02155.pdf.
2. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 120.
3. Chinyere Onwubiko, *An Introduction to Engineering*, Mission, Kansas: Schroff Development Corporation, 1997, page 205-206.

Examples

```
library(iemisc)

# Compound Interest example from SFPE Reference text
# Modified example to provide the compounded interest amount paid only

CompIntCharg(100, 5, 10, frequency = "annual") # the interest rate is 10\%
```

CompIntPaid *Compound Interest Paid (Engineering Economics)*

Description

Computes the total amount paid at the end of n periods using compound interest

Usage

```
CompIntPaid(
  P,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

`P`	numeric vector that contains the present value(s)
`n`	numeric vector that contains the period value(s)
`i`	numeric vector that contains the interest rate(s) as a percent
`frequency`	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

Compound Interest is expressed as

$$S_n = P\left(1 + i\right)^n$$

P the "principal amount (lent or borrowed)"

S_n the "total amount paid back"

i the "interest rate per interest period"

n the "number of interest periods"

Value

CompIntPaid numeric vector that contains the total amount paid at the end of n periods rounded to 2 decimal places

Author(s)

Irucka Embry

References

1. *SFPE Handbook of Fire Protection Engineering*. 3rd Edition, DiNenno, P. J.; Drysdale, D.; Beyler, C. L.; Walton, W. D., Editor(s), page 5-94, 2002. Chapter 7; Section 5; NFPA HFPE-02. See https://web.archive.org/web/20180127185316/http://fire.nist.gov/bfrlpubs/build02/PDF/b02155.pdf.
2. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 120.
3. Chinyere Onwubiko, *An Introduction to Engineering*, Mission, Kansas: Schroff Development Corporation, 1997, pages 205-206.

Examples

```
library(iemisc)

# Compound Interest example from SFPE Reference text
CompIntPaid(100, 5, 10, frequency = "annual") # the interest rate is 10%
```

---

concr_mix_lightweight_strength

Concrete Mix Design for Structural Lightweight Concrete

Description

Calculates the amount of cement, sand, gravel, and water needed for a test batch volume of structural lightweight concrete using the weight method. Note: Currently, this function only works with air-entrained concrete as the author has not found a table to compute the weight of concrete for nonair-entrained concrete.

Usage

```
concr_mix_lightweight_strength(
  fc,
  slump_use = NULL,
  max_size_aggr,
  FM,
  sgf_coarse,
  dry_rod_wt_aggr,
  absorp_coarse,
  absorp_fine,
  entrainment = c("Nonair", "Air"),
 construction_type = c("Beams and reinforced walls", "Building columns", "Floor slabs"),
  slump_value = c("Maximum", "Maximum + 1", "Minimum", "Minimum + 1"),
  exposure = c("Mild", "Moderate", "Extreme"),
```

```
    structure_type = c("Thin section", "Other"),
    severe_exposure = c("Wet", "Sea water"),
   trial_batch = c("1 cubic yard", "1 cubic foot", "0.5 cubic foot", "0.2 cubic foot",
     "All")
)
```

Arguments

`fc`	numeric vector that contains the concrete compressive strength (psi)
`slump_use`	numeric vector that contains the amount of slump (in)
`max_size_aggr`	numeric vector that contains the maximum aggregate size (in)
`FM`	numeric vector that contains the "Fineness Modulus of sand" (dimensionless)
`sgf_coarse`	numeric vector that contains the "specific gravity factor" of the coarse aggregate (dimensionless)
`dry_rod_wt_aggr`	numeric vector that contains the dry rodded weight of aggregate ["oven-dry loose weight of coarse aggregate"] (lb/ft^3)
`absorp_coarse`	numeric vector that contains the absorption of the coarse aggregate (whole number percent)
`absorp_fine`	numeric vector that contains the absorption of the fine aggregate (whole number percent)
`entrainment`	character vector that contains either Air or Nonair entrainment
`construction_type`	character vector that contains the intended type of construction
`slump_value`	character vector that contains the slump value (Maximum, Maximum + 1, Minimum, or Minimum + 1). It is "+ 1 in. for methods of consolidation other than vibration"
`exposure`	character vector that contains the exposure value (Mild, Moderate, or Extreme) for use with Air entrained concrete mixes
`structure_type`	character vector that contains the severe exposure value ["Thin sections (railings, curbs, sills, ledges, ornamental work) and sections with less than 1 in. cover over steel" or "All other structures"] for use with Air entrained concrete mixes with severe exposure
`severe_exposure`	character vector that contains the severe exposure value ("Structure wet continuously or frequently and exposed to freezing and thawing" or "Structure exposed to sea water or sulfates") for use with Air entrained concrete mixes with severe exposure
`trial_batch`	character vector that contains the volume of the trial batch mix to return (1 cubic yard, 1 cubic foot, 0.5 cubic foot, 0.2 cubic foot, or All)

Value

the amounts of cement, sand, gravel, and water in lb, rounded to the hundredth, as a `list` to make 1 yd^3, 1 ft^3, 0.5 ft^3, or 0.2 ft^3 of structural lightweight concrete or as a `data.table` containing all batch volumes.

Author(s)

Irucka Embry, Hans Werner Borchers for the interp1 and interp2 functions from pracma

Source

1. r - Error when doing bilinear interpolation with 'interp2 pracma'; any better way for 2D interpolation? - Stack Overflow answered and edited by Zheyuan Li on Dec 8 2016. See https://stackoverflow.com/questions/41032225/error-when-doing-bilinear-interpolation-with-interp2-prac
2. r - data.table 1.10.0 - why does a named column index value not work while a integer column index value works without with = FALSE - Stack Overflow answered and edited by Matt Dowle on Dec 8 2016. See https://stackoverflow.com/questions/41032225/error-when-doing-bilinear-interpolation-with-interp2-pracma-any-better-way.

References

ACI Committee 211, *Standard Practice for Selecting Proportions for Structural Lightweight Concrete (ACI 211.2-98)*, American Concrete Institute, Farmington Hills, MI, 18 pages. 1998, 211.2-98.

See Also

concr_mix_normal_strength for Concrete Mix Design for Normal Strength (Normal-weight) Concrete

Examples

```
library(iemisc)

# Example A from Section 3.2.3 using the 'Weight method (specific gravity
# pycnometers)' from ACI Committee 211
# Design a concrete mix for 3500 psi concrete strength, 'floor slab of a
# multistory structure subjected to freezing and thawing during
# construction', and a maximum size of aggregate = 3/4 in, with Fineness
# Modulus of sand = 2.80, 'the oven-dry loose weight of coarse aggregate' =
# 47 lb/ft^3 with a specific gravity factor = 1.50, and a absorption of
# 11.0\% for the coarse aggregate and 1.0\% for the fine aggregate.

concr_mix_lightweight_strength(fc = 3500, max_size_aggr = 3 / 4, FM = 2.80,
sgf_coarse = 1.50, dry_rod_wt_aggr = 47, absorp_coarse = 11.0,
absorp_fine = 1.0, entrainment = "Air", construction_type = "Floor slabs",
slump_value = "Maximum", exposure = "Extreme", structure_type = "Other",
severe_exposure = "Wet", trial_batch = "1 cubic foot")
```

---

concr_mix_normal_strength
Concrete Mix Design for Normal Strength (Normal-weight) Concrete

---

Description

Calculates the amount of cement, sand, gravel, and water needed for a test batch volume of normal strength concrete using the volumetric method.

Usage

```
concr_mix_normal_strength(
  fc,
  slump_use = NULL,
  max_size_aggr,
  FM,
  dry_rod_wt_aggr,
  mc_coarse,
  mc_fine,
  entrainment = c("Nonair", "Air"),
  construction_type = c("Reinforced Foundation walls and footings",
    "Plain footings and caissons", "Slabs, beams and reinforced walls",
    "Building Columns", "Pavements and slabs", "Heavy mass construction"),
  slump_value = c("Maximum", "Maximum + 1", "Minimum"),
  exposure = c("Nonair", "Mild", "Moderate", "Extreme"),
 trial_batch = c("1 cubic yard", "1 cubic foot", "0.5 cubic foot", "0.2 cubic foot",
    "All")
)
```

Arguments

fc	numeric vector that contains the concrete compressive strength (psi)
slump_use	numeric vector that contains the amount of slump (in)
max_size_aggr	numeric vector that contains the maximum aggregate size (in)
FM	numeric vector that contains the "Fineness Modulus of sand" (dimensionless)
dry_rod_wt_aggr	numeric vector that contains the dry rodded weight of aggregate (lb/ft^3)
mc_coarse	numeric vector that contains the moisture content of the coarse aggregate (whole number percent)
mc_fine	numeric vector that contains the moisture content of the fine aggregate (whole number percent)
entrainment	character vector that contains either Air or Nonair entrainment
construction_type	character vector that contains the intended type of construction

`slump_value` character vector that contains the slump value (Maximum, Maximum + 1, or Minimum). It is "+ 1 in. for methods of consolidation other than vibration"

`exposure` character vector that contains the exposure value (Mild, Moderate, or Extreme) for use with Air entrained concrete mixes or Nonair to indicate that it is a Nonair entrained concrete mix

`trial_batch` character vector that contains the volume of the trial batch mix to return (1 cubic yard, 1 cubic foot, 0.5 cubic foot, 0.2 cubic foot, or All)

Value

the amounts of cement, sand, gravel, and water in lb, rounded to the hundredth, as a `list` to make 1 yd^3, 1 ft^3, 0.5 ft^3, or 0.2 ft^3 of normal strength concrete or as a `data.table` containing all batch volumes.

Author(s)

Irucka Embry, Hans Werner Borchers for the interp1 and interp2 functions from pracma

Source

1. r - Error when doing bilinear interpolation with 'interp2 pracma'; any better way for 2D interpolation? - Stack Overflow answered and edited by Zheyuan Li on Dec 8 2016. See https://stackoverflow.com/questions/41032225/error-when-doing-bilinear-interpolation-with-interp2-prac
2. r - data.table 1.10.0 - why does a named column index value not work while a integer column index value works without with = FALSE - Stack Overflow answered and edited by Matt Dowle on Dec 8 2016. See https://stackoverflow.com/questions/41032225/error-when-doing-bilinear-interpolation-with-interp2-pracma-any-better-way.

References

Edward G. Nawy, *Reinforced Concrete: A Fundamental Approach*, 5th Edition, Upper Saddle River, New Jersey: Pearson Prentice Hall, 2005, page 23-28.

See Also

`concr_mix_lightweight_strength` for Concrete Mix Design for Structural Lightweight Concrete

Examples

```
library(iemisc)

# 'Example 3.1 Mixture Design of Normal-weight Concrete' from Nawy
# (page 23-28)
# Design a concrete mix for 4000 psi concrete strength, beam, and a maximum
# size of aggregate = 3/4 in, with Fineness Modulus of sand = 2.6, the dry
# rodded weight of aggregate = 100 lb/ft^3^, and a moisture content of 3\%
# for the coarse aggregate and 2\% for the fine aggregate.

concr_mix_normal_strength(fc = 4000, max_size_aggr = 3 / 4, FM = 2.6,
dry_rod_wt_aggr = 100, mc_coarse = 3, mc_fine = 2, entrainment = "Nonair",
```

```
construction_type = "Reinforced Foundation walls and footings", slump_value
= "Maximum", exposure = "Nonair", trial_batch = "1 cubic yard")
```

---

construction_decimal *Construction Decimal*

Description

Convert a construction measurement in US Customary Units (foot + inch) with or without a fraction into its equivalent as a decimal

Usage

```
construction_decimal(
  measurement,
  result = c("traditional", "librecad"),
  output = c("vector", "table")
)
```

Arguments

`measurement`	character vector that contains the construction measurement (foot + inch)
`result`	character vector that contains the decimal type [options are traditional (ex. 1.203125 = 1'-2 7/16\"] where the whole number is the value in ft and the decimal is the value in inches & librecad (ex. 14.43112 = 1'-2 7/16\"), whereby LibreCAD defines its decimal unit as "integer part separated from the fractional part of a number by a decimal". Thus, both the whole number and the decimal is the value in inches.
`output`	character vector that contains the type of output. The options are vector (just the single value as a decimal) and table [the decimal value in inch (in), feet (ft), yard (yd), millimeters (mm), centimeters (cm), and meters (m)].

Value

the construction measurement value as a numeric `vector` as a decimal or as a table (depends on the output parameters)

Note

If you only have a measurement in inches, then use `frac_to_numeric` instead.

Author(s)

Irucka Embry

Source

1. removing all non-numeric characters from a string, but not "." - R help on nabble.com answered by David Winsemius on Jul 26, 2016. See https://web.archive.org/web/20190730141421/http://r.789695.n4.nabble.com/removing-all-non-numeric-characters-from-a-string-but-not-quot-c html. Retrieved thanks to the Internet Archive: Wayback Machine
2. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
3. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-askec
4. regex - Replace single backslash in R - Stack Overflow answered and edited by Hong Ooi on Aug 21, 2014. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/25424382/replace-single-backslash-in-r.

References

1. LibreCAD v2.2.0 - User Manual - Fundamentals: Units, 7 May 2022, https://librecad-docs-dev.readthedocs.io/en/latest/ref/fundamentals.html#units.
2. Spike, 1 January 2022, "Foot and Inch to Decimal Format Conversion", https://www.spikevm.com/calculators/fraction-decimal-calculators.php.

Examples

```
# Please refer to the iemisc: Construction Measurement Examples vignette for
# additional examples

# Example 1

library(iemisc)

construction_decimal("2'-0\"", result = "traditional", output = "vector")

construction_decimal("1'-2 7/16\"", result = "librecad", output = "vector")



# Example 2

library(iemisc)

construction_decimal("0 6", result = "traditional", output = "vector")
# read as 0 feet 6 inches

construction_decimal("0 6", result = "librecad", output = "vector")
# read as 0 feet 6 inches


# Example 3
```

```
library(iemisc)

tss1 <- "48'-0 1/2\""
tss2 <- "56-9 1/2\""

sum(construction_decimal(tss1, result = "traditional", output = "vector"),
construction_decimal(tss2, result = "traditional", output = "vector"))



# See Source 2 and Source 3

# Example 4

library(iemisc)

try(construction_decimal(5, result = "traditional", output =
"vector")) # please see the error message

ex_error <- character(0)
try(construction_decimal(ex_error, result = "traditional",
output = "vector")) # please see the error message

try(construction_decimal(NA, result = "traditional", output =
"vector")) # please see the error message

try(construction_decimal("feet", result = "traditional", output =
"vector")) # please see the error message




# Example 5

library(iemisc)

app1 <- "5' 2\""

app2 <- "6' 3\""

app3 <- construction_decimal(app1, result = "traditional", output = "vector") *
construction_decimal(app2, result = "traditional", output = "vector")
app3

# If you want to have the fractional value using 16ths, do the following:

construction_fraction(app3, type = "traditional", result = "traditional",
fraction = 16)
```

construction_decimal_eng
Construction Decimal Engineering (LibreCAD Style)

Description

Convert a construction measurement in US Customary Units (foot + inch) with or without a fraction into its equivalent as an Engineering value (LibreCAD style)

Usage

```
construction_decimal_eng(measurement)
```

Arguments

`measurement`	character or numeric vector that contains the construction measurement (decimal or foot + inch)

Value

the engineering construction measurement value as a character `vector`. In LibreCAD, a construction measurement of 1'-2 7/16" = 14.43112 (decimal) = 1'-2.43112" (engineering).

Author(s)

Irucka Embry, R. van Twisk (rs_units.cpp code)

Source

regex - Replace single backslash in R - Stack Overflow answered and edited by Hong Ooi on Aug 21, 2014. (Also see the additional comments in response to the answer.) See `https://stackoverflow.com/questions/25424382/replace-single-backslash-in-r`.

References

1. LibreCAD, User Manual - Fundamentals: Units: Engineering and Decimal, 7 May 2022, `https://librecad-docs-dev.readthedocs.io/en/latest/ref/fundamentals.html#units`.
2. LibreCAD rs_units.cpp code referencing how to calculate the engineering units.

Examples

```
# Please refer to the iemisc: Construction Measurement Examples vignette for
# additional examples

library(iemisc)

librecad1a <- "1 ft 2 7/16\""

construction_decimal_eng(librecad1a)


librecad4a <- 14.43112

construction_decimal_eng(librecad4a)


librecad5a <- 14.4375

construction_decimal_eng(librecad5a)


librecad6a <- 17.71354

construction_decimal_eng(librecad6a)


librecad7a <- 86.000000

construction_decimal_eng(librecad7a)


checkst <- 14.43112

construction_decimal_eng(checkst)

construction_fraction(checkst, type = "librecad", result = "traditional",
fraction = 16)
```

construction_fraction *Construction Fraction*

Description

Convert a construction measurement in US Customary Units as a decimal into its nearest equivalent as foot + inch with or without a fraction

Usage

```
construction_fraction(
  measurement,
  type = c("traditional", "librecad"),
  result = c("traditional", "inch"),
  fraction = c(0, 2, 4, 8, 16, 32, 64, 100, 128, 256)
)
```

Arguments

`measurement`	numeric vector that contains the construction measurement as a decimal
`type`	character vector that contains the decimal type [options are traditional (ex. 1.203125 = 1'-2 7/16\"] where the whole number is the value in ft and the decimal is the value in inches & librecad (ex. 14.4375 = 1'-2 7/16\"), whereby LibreCAD defines its decimal unit as "integer part separated from the fractional part of a number by a decimal". Thus, both the whole number and the decimal is the value in inches.
`result`	character vector that contains the resulting fraction type (options are traditional (ex. 1.203125 = 1 ft 2 7/16 in) & inch (ex. 14.4375 = 14 7/16 in)). This is the same as fractional (fractional inch) in LibreCAD.
`fraction`	numeric vector that contains the fractional part to return. The options are 0, 2, 4, 8, 16, 32, 64, 100, 128, or 256.

Value

the construction measurement value as a character `vector` as foot + fraction of an inch or as inch + fraction of an inch (depends on the parameters)

Author(s)

Irucka Embry

References

1. Spike, 7 May 2022, "How to Convert Feet in Decimal Format to Foot, Inch and Fraction Values", https://www.spikevm.com/construction-math/convert-decimal-fraction.php.
2. myCarpentry, 7 May 2022, "Online Fraction Calculator", https://www.mycarpentry.com/online-fraction-calculator.html.
3. LibreCAD, User Manual - Fundamentals: Units: Architectural and Decimal, 7 May 2022, https://librecad-docs-dev.readthedocs.io/en/latest/ref/fundamentals.html#units.
4. Inch Calculator. Inch Fraction Calculator – Convert Decimal to Inches, 9 May 2022, https://www.inchcalculator.com/inch-fraction-calculator/.

Examples

```
# Please refer to the iemisc: Construction Measurement Examples vignette for
# additional examples

library(iemisc)

# Example 1 from the Spike Reference

check1 <- 18.649 # decimal feet

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 16)

# Reverse the calculation to check out the absolute error

check2 <- construction_decimal(construction_fraction(check1,
type = "traditional", result = "traditional", fraction = 16),
result = "traditional", output = "vector")

fracture::fracture(check2 - check1) # difference in inches

# by approximate error

approxerror(check2, check1) # answer as a percent (\%)


# check all other fraction levels

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 0)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 2)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 4)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 8)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 32)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 64)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 100)

construction_fraction(check1, type = "traditional", result =
"traditional", fraction = 128)
```

```
# Example 2

library(iemisc)
import::from(fpCompare, "%==%")


x1 <- construction_fraction(1.203125, type = "traditional", result =
"traditional", fraction = 16)

x2 <- construction_fraction(14.4375, type = "librecad", result =
"inch", fraction = 16)

x3 <- construction_fraction(14.4375, type = "librecad", result =
"traditional", fraction = 16)

x4 <- construction_fraction(14.43112, type = "librecad", result =
"traditional", fraction = 16)

x5 <- construction_fraction(14.43112, type = "librecad", result =
"inch", fraction = 16)


ex1 <- frac_to_numeric(x2)

ex2 <- construction_decimal(x1, result = "librecad", output = "vector")

ex3 <- construction_decimal(x3, result = "librecad", output = "vector")

ex4 <- construction_decimal(x4, result = "librecad", output = "vector")

ex5 <- frac_to_numeric(x5)


# check if ex1, ex2, ex3, ex4, and ex5 are equivalent
ex1 %==% ex2

ex1 %==% ex3

ex1 %==% ex4

ex1 %==% ex5

ex2 %==% ex3

ex2 %==% ex4

ex2 %==% ex5

ex3 %==% ex4
```

```
ex3 %==% ex5

ex4 %==% ex5



# Example 3 (from the Inch Calculator Reference)

library(iemisc)

construction_fraction(2.695, type = "librecad", result = "traditional",
fraction = 16)

construction_fraction(2.695, type = "librecad", result = "inch",
fraction = 16)



# Example 4

library(iemisc)

construction_fraction(17.71354, type = "traditional", result = "traditional",
fraction = 16)

construction_fraction(17.71354, type = "traditional", result = "inch",
fraction = 16)
```

cosd *Cosine (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of cosine for each element of x in degrees in a manner compatible with GNU Octave/MATLAB. Zero is returned for any 'elements where (x - 90) / 180 is an integer.' Reference: Eaton.

Usage

```
cosd(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The cosine of each element of x in degrees. Zero for any 'elements where (x - 90) / 180 is an integer.'

Note

Note: If you have a radian (rad) angle value, use cos instead.

Author(s)

David Bateman (GNU Octave cosd), Irucka Embry

Source

1. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
2. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-asked

References

1. John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 553.
2. Wikimedia Foundation, Inc. Wikipedia, 24 February 2019, "Radian", https://en.wikipedia.org/wiki/Radian.

Examples

```
library(iemisc)

# Example from GNU Octave cosd
```

```
cosd(seq(0, 80, by = 10))


# See Source 1 and Source 2

library(iemisc)

try(cosd("90"))
```

---

cotd *Cotangent (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of inverse secant for each element of x in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
cotd(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The inverse secant of each element of x in degrees.

Author(s)

David Bateman (GNU Octave cotd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave cotd
cotd (seq(0, 80, by = 10))

cotd (c(0, 180, 360))

cotd (c(90, 270))
```

---

cscd *Cosecant (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of cosecant for each element of x in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
cscd(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The cosecant of each element of x in degrees.

Note

Note: If you have a radian (rad) angle value, use asin instead.

Author(s)

David Bateman (GNU Octave cscd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 554.

Examples

```
library(iemisc)

# Examples from GNU Octave cscd
cscd (seq(0, 90, by = 10))

cscd (c(0, 180, 360))

cscd (c(90, 270))
```

---

cv *Coefficient of variation (CV)*

Description

This function computes the sample coefficient of variation (CV).

Usage

```
cv(x, na.rm = FALSE)
```

Arguments

x	numeric vector, matrix, data.frame, or data.table that contains the sample data points.
na.rm	logical vector that determines whether the missing values should be removed or not.

Details

CV is expressed as

$$\frac{s}{\bar{x}} \cdot 100$$

s the sample standard deviation

$\bar{x}$ the sample arithmetic mean

Value

coefficient of variation (CV), as a percent (%), as an R object: a numeric `vector` or a named numeric vector if using a named object (`matrix`, `data.frame`, or `data.table`). The default choice is that any NA values will be kept (`na.rm = FALSE`). This can be changed by specifying `na.rm = TRUE`, such as `cv(x, na.rm = TRUE)`.

Author(s)

Irucka Embry

Source

1. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
2. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-askec

References

1. Masoud Olia, Ph.D., P.E. and Contributing Authors, *Barron's FE (Fundamentals of Engineering Exam)*, 3rd Edition, Hauppauge, New York: Barron's Educational Series, Inc., 2015, page 84.
2. Irwin R. Miller, John E. Freund, and Richard Johnson, *Probability and Statistics for Engineers*, Fourth Edition, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1990, page 25, 38.

See Also

sgm for geometric mean, shm for harmonic mean, rms for root-mean-square (RMS), relerror for relative error, approxerror for approximate error, and ranges for sample range.

Examples

```
# Example 2.60 from Miller (page 38)

library(iemisc)

x <- c(14, 12, 21, 28, 30, 63, 29, 63, 55, 19, 20)
# suspended solids in parts per million (ppm)

cv(x)


# using a matrix of the numeric vector x
mat1 <- matrix(data = x, nrow = length(x), ncol = 1, byrow = FALSE,
        dimnames = list(c(rep("", length(x))), "Samples"))
cv(mat1)


# using a data.frame of the numeric vector x
df <- data.frame(x)
cv(df)


# using a data.table of the numeric vector x
```

```
library("data.table")

dt <- data.table(x)
cv(dt)



# modified Example 2.60 from Miller (page 38)
xx <- c(14, 12, 21, 28, 30, 63, 29, 63, 55, 19, 20, NA)
# suspended solids in parts per million (ppm)

cv(xx) # na.rm = FALSE is the default
cv(xx, na.rm = TRUE)





# See Source 1 and Source 2

# Example 3

# Please see the error messages

library(iemisc)

try(cv(0))
try(cv(1))
try(cv(1003.23))






# Example 4 - from the archived cvcqv README

xu <- c(0.2, 0.5, 1.1, 1.4, 1.8, 2.3, 2.5, 2.7, 3.5, 4.4, 4.6, 5.4,
5.4, 5.7, 5.8, 5.9, 6.0, 6.6, 7.1, 7.9)

results2 <- cv(xu)
results2
```

c_composite_CN *Composite CN (Curve Number) with Connected Impervious Area*

Description

This function computes the composite CN (Curve Number) for connected impervious areas.

Usage

```
c_composite_CN(pervious_CN, impervious)
```

Arguments

`pervious_CN`	numeric vector containing the pervious runoff curve number
`impervious`	numeric vector containing the percent imperviousness

Value

the Composite Runoff Curve Number as a single numeric vector, in the range [0, 100]

Note

Note: Please refer to iemiscdata: Weighted CN Calculations Using the Composite CN vignette in the iemiscdata package

Author(s)

Irucka Embry

References

United States Department of Agriculture Natural Resources Conservation Service Conservation Engineering Division, "Urban Hydrology for Small Watersheds Technical Release 55 (TR-55)", June 1986, pages 2-11 - 2-16, https://web.archive.org/web/20230810204711/https://directives.sc.egov.usda.gov/OpenNonWebContent.aspx?content=22162.wba [Recovered with the Internet Archive: Wayback Machine]

`density_water` *Density of Saturated Liquid Water*

Description

This function solves for the density of water using only the temperature of the water in either units of degrees Celsius, degrees Fahrenheit, or Kelvin.

Usage

```
density_water(
  Temp,
  units = c("SI", "Eng", "Absolute"),
  Eng_units = c("slug/ft^3", "lbm/ft^3")
)
```

Arguments

`Temp`	numeric vector that contains the temperature (degrees Celsius, degrees Fahrenheit, or Kelvin)
`units`	character vector that contains the system of units [options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), or `Absolute` for Absolute Units]
`Eng_units`	character vector that contains the unit for the density of water [options are slug/ft^3 or lbm/ft^3]

Details

The simplified equation is expressed as

$$\frac{rho^t}{rho_c} = 1 + b_1 \times tau^{1/3} + b_2 \times tau^{2/3} + b_3 \times tau^{5/3} + b_4 \times tau^{16/3} + b_5 \times tau^{43/3} + b_6 \times tau^{110/3}$$

where

$$rho_c = 322 \frac{kg}{m^3}$$

where

$$tau = 1 - theta$$

where

$$theta = \frac{T}{T_c}$$

where

$$T_c = 647.096K$$

with

$$b_1 = 1.99274064$$
$$b_2 = 1.09965342$$
$$b_3 = -0.510839303$$
$$b_4 = -1.75493479$$
$$b_5 = -45.5170352$$
$$b_6 = -6.74694450 * 10^5$$

\rho' = rho^t in the equation Water Density (mass divided by volume) [kg/m^3, slug/ft^3, or lbm/ft^3]

\rho_c Water Density at the critical point, kg/m^3

T the water temperature, Kelvin

T_c the critical water temperature, Kelvin

Value

the density as a numeric vector. The units are not returned.

Note

Note: 1 lbf = 1 slug * 1 ft/sec^2, thus 1 slug = 1 lbf * sec^2 / 1 ft (Reference 2)

Thus, lbm/ft^3 = lbf*s^2/ft/ft^3

Author(s)

Irucka Embry

References

IAPWS SR1-86 (1992). "Revised Supplementary Release on Saturation Properties of Ordinary Water Substance". September 1992, http://www.iapws.org/relguide/Supp-sat.html

Examples

```
# Example 1 (Compare to reference standard in Reference paper)

library(iemisc)

273.16 # K

373.1243 # K

647.096 # K

Temp <- c(273.16, 373.1243, 647.096)

round::round_r3(density_water(Temp, units = "Absolute"), d = 3)


# Reference standard

999.789 # kg/m^3
958.365 # kg/m^3
322 # kg/m^3





# Example 2 - Example from the hydraulics package

library(iemisc)

rho <- hydraulics::dens(T = 25, units = "SI"); rho

rho2 <- density_water(Temp = 25, units = "SI"); rho2
```

```
# Example 3 - compare with densityH2Ov from aiRthermo

install.load::load_package("iemisc", "units")

Temp <- 180

# create a numeric vector with the units of degrees Celsius
T_C <- set_units(Temp, "degree_C")
T_C

# create a numeric vector to convert from degrees Celsius to Kelvin
T_K <- T_C
T_K

# create a numeric vector with the units of Kelvin
units(T_K) <- make_units(K)

pre <- aiRthermo::saturation_pressure_H2O(drop_units(T_K))
pre

rho_h2o <- aiRthermo::densityH2Ov(pre, drop_units(T_K), consts =
aiRthermo::export_constants()); rho_h2o

# Should not be the same as aiRthermo deals with water vapor rather than
# saturated liquid water

density_water(Temp = drop_units(T_K), units = "Absolute")
```

`dyn_visc_water` *Absolute or Dynamic Viscosity for Liquid Water*

Description

This function solves for the absolute or dynamic viscosity of water using only the temperature of the water in either units of degrees Celsius, degrees Fahrenheit, or Kelvin.

Usage

```
dyn_visc_water(
  Temp,
```

```
  units = c("SI", "Eng", "Absolute"),
  Eng_units = c("slug/ft/s", "lbf*s/ft^2")
)
```

Arguments

`Temp` numeric vector that contains the temperature (degrees Celsius, degrees Fahrenheit, or Kelvin)

`units` character vector that contains the system of units [options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), or `Absolute` for Absolute Units]

`Eng_units` character vector that contains the unit for the dynamic viscosity of water in the English system [options are slug/ft/s or lbf*s/ft^2]

Details

The simplified equation is expressed as

$$mu_s = \frac{1}{a + bT + cT^2 + dT^3}$$

with

$$a = 557.82468$$

$$b = 19.408782$$

$$c = 0.1360459$$

$$d = -3.1160832 * 10^-4$$

\mu_s Water Absolute or Dynamic Viscosity (kg/m*s, slug/ft/s, or lbf*s/ft^2)

T the water temperature, degrees Celsius

Value

the absolute or dynamic viscosity as a numeric vector. The units are not returned.

Author(s)

Irucka Embry

References

C. O. Popiel & J. Wojtkowiak (1998). "Simple Formulas for Thermophysical Properties of Liquid Water for Heat Transfer Calculations (from 0C to 150C)". *Heat Transfer Engineering*, 19:3, 87-101, article from ResearchGate: https://www.researchgate.net/publication/239243539_Simple_Formulas_for_Thermophysical_Properties_of_Liquid_Water_for_Heat_Transfer_Calculations_from_0C_to_150C.

Examples

```
# Example 1 (Compare to the tabulated values in the Reference paper)

install.load::load_package("iemisc", "data.table", "round")

Temp <- c(0, 0.01, 3.86, seq(5, 95, by = 5), 99.974, seq(100, 150, by = 5))

dynamic_viscosity <- data.table("Temperature (degrees C)" = Temp,
"mu (* 10 ^ 6, kg / m*s)" = round_r3(dyn_visc_water(Temp, units = "SI")
* 10^6, d = 1))
dynamic_viscosity




# Example 2 - Example from the hydraulics package

library(iemisc)

mu <- hydraulics::dvisc(T = 55, units = "Eng"); mu

mu2 <- dyn_visc_water(Temp = 55, units = "Eng", Eng_units = "lbf*s/ft^2"); mu2
```

`EffInt` *Effective Interest rate (Engineering Economics)*

Description

Computes the effective interest rate given the nominal interest rate per period

Usage

```
EffInt(
  r,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

`r`	numeric vector that contains the nominal interest rate(s) per period as a percent
`frequency`	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

i is expressed as

$$i = \left(1 + \frac{r}{n}\right)^n - 1$$

i the "effective interest rate per interest period"

r the "nominal interest rate"

n the "number of compounding periods per year"

Value

EffInt numeric vector that contains the effective interest rate rounded to 2 decimal places (this is the `i` used in the other Engineering Economics functions)

Author(s)

Irucka Embry

References

1. *SFPE Handbook of Fire Protection Engineering*. 3rd Edition, DiNenno, P. J.; Drysdale, D.; Beyler, C. L.; Walton, W. D., Editor(s), page 5-95, 2002. Chapter 7; Section 5; NFPA HFPE-02. See https://web.archive.org/web/20180127185316/http://fire.nist.gov/bfrlpubs/build02/PDF/b02155.pdf.
2. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, pages 164-165.

Examples

```
library(iemisc)

# Example 4-28 from Sullivan Reference text (page 165)
EffInt(1.375, frequency = "month")
# the nominal interest rate per period (month) is 1.375\%


# Example from SFPE Reference text
EffInt(18 / 12, frequency = "month")
# the nominal interest rate is 18\% per year or 18\% / 12 months
```

engr_survey — *Conversion of Engineering Survey Measurements to Decimal Degrees (KY and TN)*

Description

Takes Kentucky or Tennessee-based Northing and Easting engineering survey measurements [based in the State Plane Coordinate System (SPCS)] in meters, international foot, or US survey foot and converts those values into geodetic coordinates of the World Geodetic System (WGS) (19)84 (EPSG:4326). [MapTiler Reference] Each latitude [Y] and longitude [X] point is verified to be located within Kentucky or Tennessee.

Usage

```
engr_survey(
  Northing,
  Easting,
  units = c("survey_ft", "foot", "meters"),
  location = c("KY", "TN"),
  output = c("basic", "table"),
  utm = c(0, 1)
)
```

Arguments

`Northing`	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Northing engineering survey measurement in meters, international foot, or US survey foot
`Easting`	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Easting engineering survey measurement in meters, international foot, or US survey foot
`units`	character vector that contains the system of units (options are `survey_ft` (United States Customary System) [US survey foot], `foot`, or `meters` (International System of Units) [meters])
`location`	character vector that contains the location name ('KY' for Kentucky or 'TN' for Tennessee)
`output`	character vector that contains basic for the default result using a simple `data.table` or table for the result as a complex `data.table`
`utm`	numeric vector that contains 0 or 1 only. 0 represents do not provide the utm coordinates and 1 is to provide the utm coordinates

Value

the projected associated latitude [Y] and longitude [X] coordinates in Decimal Degrees as a `data.table` or as an enhanced `data.table` with the latitude [Y] and longitude [X] coordinates in Decimal Degrees, Degrees Minutes, Degrees Minutes Seconds & the State Plane Northing and Easting coodinates in meters, US survey foot, and the international foot

Note

Please Note: If you have Kentucky North/South Zone survey measurements, then please use the Kentucky Geological Survey, University of Kentucky - Kentucky Single Coordinate Conversion Tool (http://kgs.uky.edu/kgsweb/CoordConversionTool.asp) instead. That tool will give you the geographic coordinates too. This R function, engr_survey will only be valid for NAD83 / Kentucky Single Zone.

Useful Tennessee reference Web site Tennessee Department of Transportation Roadway Design Survey Standards https://www.tn.gov/tdot/roadway-design/survey-standards.html

Useful Kentucky reference Web site Kentucky Transportation Cabinet Survey Coordination https://transportation.ky.gov/Highway-Design/Pages/Survey-Coordination.aspx

Author(s)

Irucka Embry

Source

1. Win-Vector Blog. John Mount, June 11, 2018, "R Tip: use isTRUE()", https://win-vector.com/2018/06/11/r-tip-use-istrue/.
2. Latitude Longitude Coordinates to State Code in R - Stack Overflow answered by Josh O'Brien on Jan 6 2012 and edited by Josh O'Brien on Jun 18, 2020. See https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r.
3. r - Convert column classes in data.table - Stack Overflow answered by Matt Dowle on Dec 27 2013. See https://stackoverflow.com/questions/7813578/convert-column-classes-in-data-table.
4. Excel vlook up function in R for data frame - Stack Overflow answered by Tyler Rinker on Apr 8 2013 and edited by Tyler Rinker on Feb 26 2014. See https://stackoverflow.com/questions/15882743/excel-vlook-up-function-in-r-for-data-frame.
5. r - Converting geo coordinates from degree to decimal - Stack Overflow answered by Robbes on Jan 3 2018 and edited by ayaio on Jan 3 2018. See https://stackoverflow.com/questions/14404596/converting-geo-coordinates-from-degree-to-decimal.
6. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
7. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-asked

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Spatial Reference, Aug. 13, 2004, "EPSG:3088: NAD83 / Kentucky Single Zone", https://spatialreference.org/ref/epsg/3088/.
3. Spatial Reference, March 7, 2000, "EPSG:32136 NAD83 / Tennessee", https://spatialreference.org/ref/epsg/32136/.

4. MapTiler Team, "EPSG:4326: WGS 84 – WGS84 - World Geodetic System 1984, used in GPS, https://epsg.io/4326.
5. Tennessee Department of Transportation Design Division, Tennessee Department of Transportation Tennessee Geodetic Reference Network (TGRN) Reference Manual Second Edition Issued, page ix, https://www.tn.gov/content/dam/tn/tdot/documents/TgrnComposite.pdf.
6. Earth Point, "State Plane Coordinate System - Convert, View on Google Earth", https://www.earthpoint.us/StatePlane.aspx.
7. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, Mid Valley Oil Rad Relay Twr designation, HA1363 PID, Grayson County Kentucky, Clarkson (1967) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=HA1363.
8. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, 2006 42 07 designation, DL4005 PID, Fayette County Kentucky, Lexington West (1993) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=DL4005.

Examples

```
# Please refer to the iemisc: Engineering Survey Examples vignette for
# additional examples

# Test 1 against TGRN Manual (Reference 5)
# using the 1983 (1995) DATUM
# GPS 1 is the station name with these coordinates
# latitude (North) = 36 22 6.43923
# longitude (West) = 82 10 46.87679

library(iemisc)

Northing_test1 <- 232489.480 # provided in TGRN Manual
Easting_test1 <- 942754.124 # provided in TGRN Manual

tgrn1 <- engr_survey(Northing_test1, Easting_test1, "meters", "TN", output =
"table", utm = 0)
tgrn1



# Test 2 against TGRN Manual (Reference 5)
# using the 1983 (1995) DATUM
# GPS 60 is the station name with these coordinates
# latitude (North) = 35 8 46.44496
# longitude (West) = 89 54 24.04763

library(iemisc)

Northing_test2 <- 97296.815 # provided in TGRN Manual
Easting_test2 <- 244089.427 # provided in TGRN Manual

tgrn2 <- engr_survey(Northing_test2, Easting_test2, "meters", "TN", output =
"table", utm = 0)
```

```
tgrn2



# Test 3 against the NGS Data sheet (Reference 7)
# using the NAD 83(1993) DATUM
# with these adjusted coordinates
# latitude (North) = 37 24 17.73330
# longitude (West) = 086 14 14.18027

library(iemisc)

# The following coordinates were computed from the latitude / longitude
# using NAD 83(1993)
Northing_test3 <- "1,119,041.443" # provided in NGS Data sheet
Easting_test3 <- "1,456,861.006" # provided in NGS Data sheet

ky1 <- engr_survey(Northing_test3, Easting_test3, "meters", "KY", output =
"table", utm = 0)
ky1



# Test 4 against the NGS Data sheet (Reference 8)
# using the NAD 83(2011) DATUM
# with these no check coordinates
# latitude (North) = 38 04 23.86331
# longitude (West) = 084 32 04.55607

library(iemisc)

# The following coordinates were computed from the latitude / longitude
# using NAD 83(2011)
Northing_test4 <- "3,671,388.47" # provided in NGS Data sheet
Easting_test4 <- "4,779,718.15" # provided in NGS Data sheet

ky2 <- engr_survey(Northing_test4, Easting_test4, "survey_ft", "KY", output =
"table", utm = 0)
ky2




# Example 1
# Kentucky (KY) Northing and Easting in US Survey foot

library(iemisc)

Northing1 <- 3807594.80077

Easting1 <- 5625162.88913

dt1 <- engr_survey(Northing = Northing1, Easting = Easting1, units =
```

```
"survey_ft", location = "KY", output = "table", utm = 1)
dt1



# Example 2
# Kentucky (KY) Northing and Easting in meters

library(iemisc)

Northing2 <- 1170338.983

Easting2 <- 1624669.125

dt2 <- engr_survey(Northing2, Easting2, "meters", "KY", output = "basic",
utm = 0)
dt2




# See Source 6 and Source 7

# Please see the error messages

library(iemisc)

# Tennessee (TN) Northing and Easting in US Survey foot
Northing3 <- c("630817.6396", "502170.6065", "562,312.2349", "574,370.7178")

Easting3 <- c("2559599.9201", "1433851.6509", "1,843,018.4099", "1,854,896.0041")


dt3 <- try(engr_survey(Northing3, Easting3, "survey_ft", "TN", output = "basic",
utm = 0))


Northing4 <- c(232489.480, 234732.431)

Easting4 <- c(942754.124, 903795.239)

dt4 <- try(engr_survey(Northing4, Easting4, "survey_ft", "TN", output =
"basic", utm = 0))
```

engr_survey2 *Calculate the Distance between Engineering Survey Points*

Description

Takes engineering survey points in various units (foot, US survey foot, meters, or kilometers) and calculates the horizontal length in various units (foot, US survey foot, US survey mile, mile, meters, or kilometers).

Usage

```
engr_survey2(
  station1,
  station2,
  station_distance = 100,
  units1 = c("foot", "survey_ft", "meters", "kilometers"),
  units2 = c("foot", "survey_ft", "survey_mile", "mile", "meters", "kilometers")
)
```

Arguments

`station1` character vector that contains the begin engineering survey station value

`station2` character vector that contains the end engineering survey station value

`station_distance` numeric vector that contains the horizontal distance between any 2 points along the survey, the default is 100 feet

`units1` character vector that contains the system of units for the `station_distance` (options are `foot`, `survey_ft` (United States Customary System) [US survey foot], `meters` for International System of Units meters, or `kilometers` for International System of Units kilometers)

`units2` character vector that contains the system of units for the horizontal length (options are `foot`, `survey_ft` (United States Customary System) [US survey foot], `survey_mile` (United States Customary System) [US survey mile], `mile`, `meters` for International System of Units meters, or `kilometers` for International System of Units kilometers)

Value

the calculated horizontal distance in the chosen unit as a numeric vector with the units attached

Author(s)

Irucka Embry

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Ben W. Lewis, PE, July 12, 2016, "Construction Plan Reading Basics & Applications Part III Typical Calculations / Quantity Take-off's: Calculate Project Length", https://web.archive.org/web/20171008121149/http://www.richlandonline.com/Portals/0/Departments/

Procurement/SLBE/Plan%20Reading%20PowerPoint%20Presentation.pdf. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.

3. Georgia Department of Transportation Skills Development Series, Revised May 1, 2008, "Basic Highway Plan Reading", https://web.archive.org/web/20220401015405/http://www.dot.ga.gov/PartnerSmart/Training/Documents/ESD/BasicHiwyPlanReading.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Please refer to the iemisc: Engineering Survey Examples vignette for
# additional examples

# Example 1

library(iemisc)

engr_survey2("395+75", "397+13", station_distance = 100, units1 = "foot",
units2 = "foot")



# Example 2

library(iemisc)

station1 <- "333+03"
station2 <- "332+94"

engr_survey2(station1, station2, units1 = "foot", units2 = "survey_mile")



# Example 3 from Lewis Reference document page 25

library(iemisc)

station3 <- "10+25.62"
station4 <- "189+45.72"

engr_survey2(station3, station4, units1 = "foot", units2 = "mile")



# Example 4 from Georgia reference page 27 (document page 43)

library(iemisc)

engr_survey2("701+50.00", "409+69.00", station_distance = 100,
units1 = "survey_ft", units2 = "foot")
```

engr_survey3 — *Calculate the Distance between Engineering Survey Points (Length or Number of Stations)*

Description

Takes engineering survey points in various units (foot, US survey foot, meters, or kilometers) and calculates the horizontal length in various units (foot, US survey foot, US survey mile, mile, meters, or kilometers).

Usage

```
engr_survey3(
  length1,
  station_distance = 100,
  units = c("foot", "survey_ft", "survey_mile", "mile", "meters", "kilometers"),
  output = c("numeric", "string")
)
```

Arguments

`length1`	character vector that contains the beginning engineering survey station value
`station_distance`	numeric vector that contains the horizontal distance between any 2 points along the survey, the default is 100 feet
`units`	character vector that contains the system of units for the `station_distance` (options are `foot`, `survey_ft` (United States Customary System) [US survey foot], `meters` for International System of Units meters, or `kilometers` for International System of Units kilometers)
`output`	character vector that contains the system of units for the horizontal length (options are `foot`, `survey_ft` (United States Customary System) [US survey foot], `survey_mile` (United States Customary System) [US survey mile], `mile`, `meters` for International System of Units meters, or `kilometers` for International System of Units kilometers)

Value

horizontal length as a numeric vector (ex. 1214.402) or as a character vector with the word stations after the number (ex. 1214.402 stations)

Author(s)

Irucka Embry

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Engineer Boards: Transportation. "Stationing. Dumb question?" Question asked by By NIKE, August 31, 2013 and answered by ptatohed on September 1, 2013. See https://engineerboards.com/threads/stationing-dumb-question.21935/.

Examples

```
# Please refer to the iemisc: Engineering Survey Examples vignette for
# additional examples

# Example 1

library(iemisc)

# "What the others said is correct. 1 station is equal to 100 feet. So when
# asked how many stations are in (3.2mi x 5280ft/mi = ) 16,896 feet, you are being
# asked how many 100 foot-segments are in 16,896 feet? The answer of course is
# 16,896ft / 100ft/sta = 168.96 sta." Source: Reference 2

length1 <- "16,896" # feet

engr_survey3(length1, station_distance = 100, units = "foot", output = "numeric")
engr_survey3(length1, station_distance = 100, units = "foot", output = "string")
# the answer provides the number of stations

# Note: Both answers should be the same as 3.2 miles = 16,896 feet.

length2 <- 3.2 # mile

engr_survey3(length2, station_distance = 100, units = "mile", output = "numeric")
engr_survey3(length2, station_distance = 100, units = "mile", output = "string")
# the answer provides the number of stations
```

---

engr_survey4 *Calculate the Station Distance between Engineering Survey Points*

Description

Takes engineering survey points in various units (foot, US survey foot, meters, or kilometers) and determines the upstream station location.

Usage

```
engr_survey4(
  object,
  ds_station,
  station_distance = 100,
  units = c("foot", "survey_ft", "survey_mile", "mile", "meters", "kilometers")
)
```

Arguments

`object`	numeric vector that contains the upstream length
`ds_station`	character vector that contains the ending engineering survey station value
`station_distance`	numeric vector that contains the horizontal distance between any 2 points along the survey, the default is 100 feet
`units`	character vector that contains the system of units for the `station_distance` (options are `foot`, `survey_ft` (United States Customary System) [US survey foot], `meters` for International System of Units meters, or `kilometers` for International System of Units kilometers)

Value

the final engineering survey position (ex. Sta. 5+55)

Author(s)

Irucka Embry

Source

r - Insert a character at a specific location in a string - Stack Overflow answered and edited by Justin on Dec 13 2012. See https://stackoverflow.com/questions/13863599/insert-a-character-at-a-specific-loca

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Engineer Boards: Transportation. "Stationing. Dumb question?" Question asked by By NIKE, August 31, 2013 and answered by ptatohed on September 1, 2013. See https://engineerboards.com/threads/stationing-dumb-question.21935/.

Examples

```
# Please refer to the iemisc: Engineering Survey Examples vignette for
# additional examples

library(iemisc)
```

```
# Example 1

# "Conversely, if you were asked at what station a manhole 555 ft upstream of
# Sta 0+00 is, your answer would be 0.00 sta + 555 ft / 100 ft/sta = 5.55 sta =
# Sta 5+55." Source: Reference 2

engr_survey4(555, "0+00", units = "foot")
```

---

engr_survey_batch *Conversion of Engineering Survey Measurements to Decimal Degrees (KY and TN) - Batch*

Description

Takes Kentucky or Tennessee-based Northing and Easting engineering survey measurements [based in the State Plane Coordinate System (SPCS)] in meters, international foot, or US survey foot and converts those values into geodetic coordinates of the World Geodetic System (WGS) (19)84 (EPSG:4326). [MapTiler Reference] Each latitude [Y] and longitude [X] point is verified to be located within Kentucky or Tennessee. This is the batch version of engr_survey as it processes multiple pairs of Northing and Easting points at a time.

Usage

```
engr_survey_batch(
  Northing,
  Easting,
  units = c("survey_ft", "foot", "meters"),
  location = c("KY", "TN"),
  output = c("basic", "table")
)
```

Arguments

Northing	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Northing engineering survey measurement in meters, international foot, or US survey foot [a minimum of 2 points each time]
Easting	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Easting engineering survey measurement in meters, international foot, or US survey foot [a minimum of 2 points each time]

units	character vector that contains the system of units (options are survey_ft (United States Customary System) [US survey foot], foot, or meters (International System of Units) [meters] [only 1 set of units at a time])
location	character vector that contains the location name ('KY' for Kentucky or 'TN' for Tennessee) [only 1 location at a time]
output	character vector that contains basic for the default result using a simple data.table or table for the result as a complex data.table [using rbindlist therefore resulting in only 1 data.table]

Value

the projected associated latitude [Y] and longitude [X] coordinates in Decimal Degrees using the sf system

Note

Please Note: If you have Kentucky North/South Zone survey measurements, then please use the Kentucky Geological Survey, University of Kentucky - Kentucky Single Coordinate Conversion Tool (http://kgs.uky.edu/kgsweb/CoordConversionTool.asp) instead. That tool will give you the geographic coordinates too. This R function, engr_survey_batch will only be valid for NAD83 / Kentucky Single Zone.

Useful Tennessee reference Web site Tennessee Department of Transportation Roadway Design Survey Standards https://www.tn.gov/tdot/roadway-design/survey-standards.html

Useful Kentucky reference Web site Kentucky Transportation Cabinet Survey Coordination https://transportation.ky.gov/Highway-Design/Pages/Survey-Coordination.aspx

Author(s)

Irucka Embry

Source

1. Win-Vector Blog. John Mount, June 11, 2018, "R Tip: use isTRUE()", https://win-vector.com/2018/06/11/r-tip-use-istrue/.
2. Latitude Longitude Coordinates to State Code in R - Stack Overflow answered by Josh O'Brien on Jan 6 2012 and edited by Josh O'Brien on Jun 18, 2020. See https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r.
3. r - Convert column classes in data.table - Stack Overflow answered by Matt Dowle on Dec 27 2013. See https://stackoverflow.com/questions/7813578/convert-column-classes-in-data-table.
4. Excel vlook up function in R for data frame - Stack Overflow answered by Tyler Rinker on Apr 8 2013 and edited by Tyler Rinker on Feb 26 2014. See https://stackoverflow.com/questions/15882743/excel-vlook-up-function-in-r-for-data-frame.
5. r - Converting geo coordinates from degree to decimal - Stack Overflow answered by Robbes on Jan 3 2018 and edited by ayaio on Jan 3 2018. See https://stackoverflow.com/questions/14404596/converting-geo-coordinates-from-degree-to-decimal.

6. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
7. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-askec

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Spatial Reference, Aug. 13, 2004, "EPSG:3088: NAD83 / Kentucky Single Zone", https://spatialreference.org/ref/epsg/3088/.
3. Spatial Reference, March 7, 2000, "EPSG:32136 NAD83 / Tennessee", https://spatialreference.org/ref/epsg/32136/.
4. MapTiler Team, "EPSG:4326: WGS 84 – WGS84 - World Geodetic System 1984, used in GPS, https://epsg.io/4326.
5. Tennessee Department of Transportation Design Division, Tennessee Department of Transportation Tennessee Geodetic Reference Network (TGRN) Reference Manual Second Edition Issued, page ix, https://www.tn.gov/content/dam/tn/tdot/documents/TgrnComposite.pdf.
6. Earth Point, "State Plane Coordinate System - Convert, View on Google Earth", https://www.earthpoint.us/StatePlane.aspx.
7. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, Mid Valley Oil Rad Relay Twr designation, HA1363 PID, Grayson County Kentucky, Clarkson (1967) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=HA1363.
8. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, 2006 42 07 designation, DL4005 PID, Fayette County Kentucky, Lexington West (1993) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=DL4005.

Examples

```
# Please refer to the iemisc: Engineering Survey Examples vignette for
# additional examples

# Example 1

# Tennessee (TN) Northing and Easting in US Survey foot

library(iemisc)

Northing1 <- c("630817.6396", "502170.6065", "562,312.2349", "574,370.7178")

Easting1 <- c("2559599.9201", "1433851.6509", "1,843,018.4099", "1,854,896.0041")

dt1 <- engr_survey_batch(Northing1, Easting1, "survey_ft", "TN", output = "basic")
```

```
# See Source 6 and Source 7

# Please see the error messages

library(iemisc)

Northing1 <- c("630817.6396", "502170.6065", "562,312.2349", "574,370.7178")

Easting1 <- c("2559599.9201", "1433851.6509", "1,843,018.4099", "1,854,896.0041")

Northing2 <- c(232489.480, 234732.431)

Easting2 <- c(942754.124, 903795.239)


dt1A <- try(engr_survey_batch(Northing1[1], Easting1[1], "survey_ft", "TN",
output = "basic"))
dt1A # first set of Northing, Easting points

dt1B <- try(engr_survey_batch(Northing1[2], Easting1[2], "survey_ft", "TN",
output = "basic"))
dt1B # second set of Northing, Easting points

dt1C <- try(engr_survey_batch(Northing1[3], Easting1[3], "survey_ft", "TN",
output = "basic"))
dt1C # third set of Northing, Easting points

dt1D <- try(engr_survey_batch(Northing1[4], Easting1[4], "survey_ft", "TN",
output = "basic"))
dt1D # fourth set of Northing, Easting points


dt4A <- try(engr_survey_batch(Northing2[1], Easting2[1], "meters", "TN",
output = "table"))
dt4A

dt4B <- try(engr_survey_batch(Northing2[2], Easting2[2], "meters", "TN",
output = "table"))
dt4B
```

`engr_survey_reverse` *Conversion of Latitude/Longitude Coordinates to Engineering Survey Measurements (KY and TN)*

Description

Takes geodetic coordinates of the World Geodetic System (WGS) (19)84 (EPSG:4326) [MapTiler Reference] and converts those values into Kentucky or Tennessee-based Northing and Easting engineering survey measurements [based in the State Plane Coordinate System (SPCS)] in meters, international foot, or US survey foot. Each latitude [Y] and longitude [X] point is verified to be located within Kentucky or Tennessee.

Usage

```
engr_survey_reverse(
  latitude,
  longitude,
  units = c("survey_ft", "foot", "meters"),
  location = c("KY", "TN"),
  output = c("basic", "table"),
  utm = c(0, 1)
)
```

Arguments

`latitude`	numeric vector [or character vector with spaces, degree symbol, single quotation mark, and/or escaped quotation mark (\")] that contains the latitude coordinate point. The following possibilities are valid: -25.02, "25\U00B056'50.2068\"N"; "15\U00B056'58.7068"; "37'1'54.3'N"; "35 8 46.44496", "35'8'46.44496". If the North designation is not provided, then it will be added. The latitude/longitude coordinate pair has to be either a numeric or character vector (no mixing).
`longitude`	numeric vector [or character vector with spaces, degree symbol, single quotation mark, and/or escaped quotation mark (\")] that contains the latitude coordinate point. The following possibilities are valid: 09.83, "25\U00B056'50.2068\"W"; "15\U00B056'58.7068"; "37'1'54.3'E"; "35 8 46.44496", "35'8'46.44496". If the West designation is not provided, then it will be added. The latitude/longitude coordinate pair has to be either a numeric or character vector (no mixing).
`units`	character vector that contains the system of units (options are `survey_ft` (United States Customary System) [US survey foot], `foot`, or `meters` (International System of Units) [meters])
`location`	character vector that contains the location name ('KY' for Kentucky or 'TN' for Tennessee)
`output`	character vector that contains basic for the default result using a simple `data.table` or table for the result as a complex `data.table`
`utm`	numeric vector that contains 0 or 1 only. 0 represents do not provide the utm coordinates and 1 is to provide the utm coordinates

Value

the geodetic coordinates as projected SPCS Northing and Easting coordinates as a data.table

Note

Please Note: If you have Kentucky North/South Zone survey measurements, then please use the Kentucky Geological Survey, University of Kentucky - Kentucky Single Coordinate Conversion Tool (http://kgs.uky.edu/kgsweb/CoordConversionTool.asp) instead. That tool will give you the geographic coordinates too. This R function, `engr_survey_reverse` will only be valid for NAD83 / Kentucky Single Zone.

Useful Tennessee reference Web site Tennessee Department of Transportation Roadway Design Survey Standards (https://www.tn.gov/tdot/roadway-design/survey-standards.html)

Useful Kentucky reference Web site Kentucky Transportation Cabinet Survey Coordination (https://transportation.ky.gov/Hig Design/Pages/Survey-Coordination.aspx)

Author(s)

Irucka Embry, Berry Boessenkool (a couple of functions are sourced from OSMscale)

Source

1. Win-Vector Blog. John Mount, June 11, 2018, "R Tip: use isTRUE()", https://win-vector.com/2018/06/11/r-tip-use-istrue/.
2. Latitude Longitude Coordinates to State Code in R - Stack Overflow answered by Josh O'Brien on Jan 6 2012 and edited by Josh O'Brien on Jun 18, 2020. See https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r.
3. r - Convert column classes in data.table - Stack Overflow answered by Matt Dowle on Dec 27 2013. See https://stackoverflow.com/questions/7813578/convert-column-classes-in-data-table.
4. Excel vlook up function in R for data frame - Stack Overflow answered by Tyler Rinker on Apr 8 2013 and edited by Tyler Rinker on Feb 26 2014. See https://stackoverflow.com/questions/15882743/excel-vlook-up-function-in-r-for-data-frame.
5. r - Converting geo coordinates from degree to decimal - Stack Overflow answered by Robbes on Jan 3 2018 and edited by ayaio on Jan 3 2018. See https://stackoverflow.com/questions/14404596/converting-geo-coordinates-from-degree-to-decimal.

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Spatial Reference, Aug. 13, 2004, "EPSG:3088: NAD83 / Kentucky Single Zone", https://spatialreference.org/ref/epsg/3088/.
3. Spatial Reference, March 7, 2000, "EPSG:32136 NAD83 / Tennessee", https://spatialreference.org/ref/epsg/32136/.
4. MapTiler Team, "EPSG:4326: WGS 84 – WGS84 - World Geodetic System 1984, used in GPS, https://epsg.io/4326.

5. Tennessee Department of Transportation Design Division, Tennessee Department of Transportation Tennessee Geodetic Reference Network (TGRN) Reference Manual Second Edition Issued, page ix, https://www.tn.gov/content/dam/tn/tdot/documents/TgrnComposite.pdf.
6. Earth Point, "State Plane Coordinate System - Convert, View on Google Earth", https://www.earthpoint.us/StatePlane.aspx.
7. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, Mid Valley Oil Rad Relay Twr designation, HA1363 PID, Grayson County Kentucky, Clarkson (1967) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=HA1363.
8. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, 2006 42 07 designation, DL4005 PID, Fayette County Kentucky, Lexington West (1993) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=DL4005.

Examples

```
# Please refer to the iemisc: Engineering Survey Examples vignette for
# additional examples

# Test against TGRN Manual (Reference 5)
# using the 1983 (1995) DATUM
# GPS 60 is the station name with these coordinates
# latitude (North) = 35 8 46.44496
# longitude (West) = 89 54 24.04763

# Northing is 97296.815 # provided in TGRN Manual
# Easting is 244089.427 # provided in TGRN Manual

library(iemisc)

latitude <- "35 8 46.44496"
longitude <- "89 54 24.04763"

Northing_test2 <- 97296.815 # provided in TGRN Manual
Easting_test2 <- 244089.427 # provided in TGRN Manual

tgrn2A <- engr_survey_reverse(latitude, longitude, "meters", "TN", output = "table",
utm = 0)
tgrn2A

tgrn2B <- engr_survey(Northing_test2, Easting_test2, "meters", "TN", output = "table",
utm = 0)
tgrn2B




# Example 1

# Tennessee

library(iemisc)
```

```
lat <- 35.8466965

long <- -88.9206794

dt1B <- engr_survey_reverse(lat, long, units = "survey_ft", location = "TN", output =
"basic", utm = 1)
dt1B




# Example 2

# Kentucky

library(iemisc)

lats <- "37'50'21.5988''N"
longs <- "84'16'12.0720'W"

dt2A <- engr_survey_reverse(lats, longs, "foot", "KY", output = "basic", utm = 0)
dt2A
```

---

f1 *Laminar Flow Equation*

Description

The Darcy friction factor (f) is used in the "Darcy equation (also known as the Weisbach equation or Darcy-Weisbach equation)" to determine the "frictional energy loss for fluids" experiencing either laminar or turbulent flow (which is based on the Reynolds number). [Reference: Lindeburg Manual]

Usage

```
f1(Re)
```

Arguments

Re	numeric vector that contains the Reynolds number (dimensionless)

Details

Calculating the Darcy friction factor (f) for pipes

Please consult the references for the equations:

f1 - laminar flow equation (References: Lindeburg Manual and Zeghadnia) f2 - Moody equation (References: Genić and Zeghadnia) f3 - Romeo, et. al. equation (References: Genić and Zeghadnia) f4 - Žarko Ćojbašića and Dejan Brkić equation (Reference: Zeghadnia) f5 - Colebrook-White equation (References: Genić and Praks) f6 - Swamee-Jaine equation (References: Genić and Zeghadnia) f7 - Zigrang-Sylvester equation (References: Genić and Zeghadnia) f8 - Vatankhah equation (Reference: Zeghadnia)

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Author(s)

Irucka Embry

References

1. Steven C. Chapra, *Applied Numerical Methods with MATLAB for Engineers and Scientists*, Second Edition, Boston, Massachusetts: McGraw-Hill, 2008, pages 157-161.
2. Didier Clamond, "Efficient resolution of the Colebrook equation", *Ind. Eng. Chem. Res.*, 2009, 48 (7), pages 3665-3671, https://arxiv.org/abs/0810.5564 and https://math.univ-cotedazur.fr/~didierc/DidPublis/ICR_2009.pdf
3. Srbislav Genić, Ivan Arandjelović, Petar Kolendić, Marko Jarić, Nikola Budimir, and Vojislav Genić, "A Review of Explicit Approximations of Colebrook's Equation", *FME (Faculty of Mechanical Engineering, Belgrade) Transactions*, 2011, 39, pages 67-71, https://www.mas.bg.ac.rs/_media/istrazivanje/fme/vol39/2/04_mjaric.pdf
4. Michael R. Lindeburg, PE, *Civil Engineering Reference Manual for the PE Exam*, Twelfth Edition, Belmont, California: Professional Publications, Inc., 2011, pages 17-5 - 17-7.
5. Michael R. Lindeburg, PE, *Practice Problems for the Civil Engineering PE Exam: A Companion to the "Civil Engineering Reference Manual"*, Twelfth Edition, Belmont, California: Professional Publications, Inc., 2011, pages 17-1 and 17-8 - 17-9.
6. Pavel Praks and Dejan Brkić, "Advanced Iterative Procedures for Solving the Implicit Colebrook Equation for Fluid Flow Friction", *Advances in Civil Engineering*, Volume 2018, Article ID 5451034, 18 pages, https://www.hindawi.com/journals/ace/2018/5451034/
7. Lotfi Zeghadnia, Jean Loup Robert, and Bachir Achour, "Explicit solutions for turbulent flow friction factor: A review, assessment and approaches classification", *Ain Shams Engineering Journal*, March 2019, Volume 10, Issue 1, pages 243-252, https://www.sciencedirect.com/science/article/pii/S2090447919300176

See Also

`Re1`, `Re2`, `Re3`, and `Re4` for the Reynolds number and `colebrook` for an accurate representation of the Colebrook-White equation

`f2`, `f3`, `f4`, `f5`, `f6`, `f7`, `f8`

Examples

```
library(iemisc)

# Examples

f1(200)

f1(1999)
```

---

f2 *Moody Equation*

Description

Moody Equation

Usage

```
f2(eps, Re, D = NULL)
```

Arguments

eps	numeric vector that contains the specific roughness for the pipe material (m or ft)
Re	numeric vector that contains the Reynolds number (dimensionless)
D	numeric vector that contains the inner diameters of the pipe (m or ft)

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f3`, `f4`, `f5`, `f6`, `f7`, `f8` f3 *Romeo, et. al. Equation*

Description

Romeo, et. al. Equation

Usage

```
f3(eps, Re, D = NULL, x0 = NULL)
```

Arguments

eps	numeric vector that contains the specific roughness for the pipe material (m or ft)
Re	numeric vector that contains the Reynolds number (dimensionless)
D	numeric vector that contains the inner diameters of the pipe (m or ft)
x0	numeric vector that contains the starting value for pracma's newtonRaphson function

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f2`, `f4`, `f5`, `f6`, `f7`, `f8` and `newtonRaphson` for `x0`

f4 *Žarko Ćojbašića and Dejan Brkić Equation*

Description

Žarko Ćojbašića and Dejan Brkić Equation

Usage

```
f4(eps, Re, D, x0 = NULL)
```

Arguments

eps	numeric vector that contains the specific roughness for the pipe material (m or ft)
Re	numeric vector that contains the Reynolds number (dimensionless)
D	numeric vector that contains the inner diameters of the pipe (m or ft)
x0	numeric vector that contains the starting value for pracma's newtonRaphson function

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f2`, `f3`, `f5`, `f6`, `f7`, `f8` and `newtonRaphson` for `x0`

f5 *Colebrook-White Equation*

Description

Colebrook-White Equation

Usage

```
f5(eps, D = NULL, Re, x0 = NULL)
```

Arguments

eps	numeric vector that contains the specific roughness for the pipe material (m or ft)
D	numeric vector that contains the inner diameters of the pipe (m or ft)
Re	numeric vector that contains the Reynolds number (dimensionless)
x0	numeric vector that contains the starting value for pracma's newtonRaphson function

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f2`, `f3`, `f4`, `f6`, `f7`, `f8` and `newtonRaphson` for `x0`

f6 *Swamee-Jaine Equation*

Description

Swamee-Jaine Equation

Usage

```
f6(eps, D = NULL, Re, x0 = NULL)
```

Arguments

`eps`	numeric vector that contains the specific roughness for the pipe material (m or ft)
`D`	numeric vector that contains the inner diameters of the pipe (m or ft)
`Re`	numeric vector that contains the Reynolds number (dimensionless)
`x0`	numeric vector that contains the starting value for pracma's newtonRaphson function

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f2`, `f3`, `f4`, `f5`, `f7`, `f8`

f7 *Zigrang-Sylvester Equation*

Description

Zigrang-Sylvester Equation

Usage

```
f7(eps, D = NULL, Re, x0 = NULL)
```

Arguments

`eps`	numeric vector that contains the specific roughness for the pipe material (m or ft)
`D`	numeric vector that contains the inner diameters of the pipe (m or ft)
`Re`	numeric vector that contains the Reynolds number (dimensionless)
`x0`	numeric vector that contains the starting value for pracma's newtonRaphson function

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f2`, `f3`, `f4`, `f5`, `f6`, `f8` and `newtonRaphson` for `x0`

f8 *Vatankhah Equation*

Description

Vatankhah Equation

Usage

```
f8(eps, D, Re)
```

Arguments

eps	numeric vector that contains the specific roughness for the pipe material (m or ft)
D	numeric vector that contains the inner diameters of the pipe (m or ft)
Re	numeric vector that contains the Reynolds number (dimensionless)

Value

the dimensionless Darcy friction factor (f) as a numeric `vector`

Note

Note: Please refer to iemisc: Calculating the Friction Loss Examples vignette for the examples

Author(s)

Irucka Embry

See Also

`f1` for the additional seealso, description, details, and references sections, `f2`, `f3`, `f4`, `f5`, `f6`, `f7`

FgivenA *Future value given Annual value (Engineering Economics)*

Description

Compute F given A

Usage

```
FgivenA(
  A,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)

FA(
  A,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

`A`	numeric vector that contains the annual value(s)
`n`	numeric vector that contains the period value(s)
`i`	numeric vector that contains the interest rate(s) as a percent
`frequency`	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

F is expressed as

$$F = A\left[\frac{(1+i)^n - 1}{i}\right]$$

F the "future equivalent"

A the "uniform series amount (occurs at the end of each interest period)"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

FgivenA numeric vector that contains the future value(s) rounded to 2 decimal places

FA data.frame of both n (0 to n) and the resulting future values rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 131-132, 142, 164.

Examples

```
library(iemisc)

# Example 4-7 from the Reference text (page 131-132)
FgivenA(23000, 40, 6, "annual") # the interest rate is 6\%

FA(23000, 40, 6, "annual") # the interest rate is 6\%
```

`FgivenAcont` *Future value given Annual value [continuous] (Engineering Economics)*

Description

Compute F given A with interest compounded continuously

Usage

```
FgivenAcont(A, n, r)
```

Arguments

`A`	numeric vector that contains the annual value(s)
`n`	numeric vector that contains the period value(s)
`r`	numeric vector that contains the continuously compounded nominal annual interest rate(s) as a percent

Details

F is expressed as

$$F = A\left[\frac{e^{rn} - 1}{e^{r} - 1}\right]$$

F the "future equivalent"

A the "annual equivalent amount (occurs at the end of each year)"

r the "nominal annual interest rate, compounded continuously"

n the "number of periods (years)"

Value

FgivenAcont numeric vector that contains the future value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 169.

Examples

```
library(iemisc)

FgivenAcont(2100, 13, 7) # the interest rate is 7%
```

FgivenP *Future value given Present value (Engineering Economics)*

Description

Compute F given P

Usage

```
FgivenP(
  P,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)

FP(
  P,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

P	numeric vector that contains the present value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as a percent
frequency	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

F is expressed as

$$F = P\left(1 + i\right)^n$$

F the "future equivalent"

P the "present equivalent"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

FgivenP numeric vector that contains the future value(s) rounded to 2 decimal places

FP data.frame of both n (0 to n) and the resulting future values rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 124, 142, 164-166.

Examples

```
library(iemisc)

# Example 4-3 from the Reference text (page 124)
FgivenP(8000, 4, 10, frequency = "annual") # the interest rate is 10\%

FP(8000, 4, 10, frequency = "annual") # the interest rate is 10\%


FgivenP(P = c(1000, 340, 23), n = c(12, 1.3, 3), i = c(10, 2, 0.3),
"annual")
# is is 10\%, 2\%, and 0.3%
# Can't use FP for this example


# Example 4-29 from the Reference text (page 165-166)
FgivenP(100, 10, 6, "quarter") # the interest rate is 6\% per quarter

FP(100, 10, 6, "quarter") # the interest rate is 6\% per quarter
```

`FgivenPcont` *Future value given Present value [continuous] (Engineering Economics)*

Description

Compute F given P with interest compounded continuously

Usage

```
FgivenPcont(P, n, r)
```

Arguments

`P`	numeric vector that contains the present value(s)
`n`	numeric vector that contains the period value(s)
`r`	numeric vector that contains the continuously compounded nominal annual interest rate(s) as a percent

Details

F is expressed as

$$F = Pe^{rn}$$

F the "future equivalent"

P the "present equivalent"

r the "nominal annual interest rate, compounded continuously"

n the "number of periods (years)"

Value

FgivenPcont numeric vector that contains the future value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 169-170.

Examples

```
library(iemisc)

# Example 4-33 from the Reference text (page 170)
FgivenPcont(10000, 2, 5) # the interest rate is 5%
```

`fractdiff` *Fractional Differences (GNU Octave/MATLAB compatible)*

Description

"Compute the fractional differences (1 - L)^d * x where L denotes the lag- operator and d is greater than -1." Source: Eaton page 853.

Usage

```
fractdiff(x, d)
```

Arguments

`x`	A numeric vector
`d`	A numeric scalar

Value

Return the fractional differences.

Note

Note: Please refer to the iemisc: Examples from GNU Octave Rem, Mod, and fractdiff Compatible Functions vignette for an example

Author(s)

Irucka Embry

Irucka Embry, GNU Octave Developers (GNU Octave code)

Source

1. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
2. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-askec

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 853.

`frac_to_numeric` *Fraction (or Mixed number) to a Decimal (Numeric Vector)*

Description

Converts a fraction or a mixed number to a decimal

Usage

```
frac_to_numeric(n)
```

Arguments

n	character vector that contains the fraction or mixed number (can also include text, ex. inch, inches, etc. that will be removed from the vector)

Value

the numeric `vector` as a decimal

Note

If you have a measurement in feet + inches, then use `construction_fraction` instead.

Author(s)

Irucka Embry

Source

removing all non-numeric characters from a string, but not "." - R help on nabble.com answered by David Winsemius on Jul 26, 2016. See https://web.archive.org/web/20190730141421/http://r.789695.n4.nabble.com/removing-all-non-numeric-characters-from-a-string-but-not-quot-quot-td472 html. Retrieved thanks to the Internet Archive: Wayback Machine.

References

1. Bill Venables, 2016-02-10, "Vulgar Fractions in R", fractional vignette, https://CRAN.R-project.org/package=fractional/vignettes/Vulgar_Fractions_in_R.html.
2. The Home Depot, 9 December 2022, "How to Read a Tape Measure", https://archive.vn/fhBmg. Provided the archive.today webpage capture for The Home Depot URL for acceptance into CRAN.
3. Wikimedia Foundation, Inc. Wikipedia, 29 December 2021, "Pi", https://en.wikipedia.org/wiki/Pi.

Examples

```
# Please refer to the iemisc: Construction Measurement Examples vignette for
# additional examples

# Example 1 -- Reference 1

library(iemisc)

xx <- as.character(fractional::fractional(1:9 / 12))

try(frac_to_numeric(xx))
# Please note that there will be an error because this function is designed to
# only process one fraction at a time.

lapply(xx, frac_to_numeric)
# Please note that this is the correct way to work with several fractions at once.




# Example 2

library(iemisc)

xi <- fracture::fracture((50:65) / 12)

try(frac_to_numeric(xi))
# Please note that there will be an error because this function is designed to
# only process one fraction at a time.

lapply(xi, frac_to_numeric)
# Please note that this is the correct way to work with several fractions at once.




# Example 3

library(iemisc)

xyy <- fracture::fracture((1:11) / 12)

try(frac_to_numeric(xyy))
# Please note that there will be an error because this function is designed to
# only process one fraction at a time.

lapply(xyy, frac_to_numeric)
# Please note that this is the correct way to work with several fractions at once.
```

```
# Example 4

library(iemisc)

xft <- as.character(MASS::fractions((1:70) / 12))

try(frac_to_numeric(xft))
# Please note that there will be an error because this function is designed to
# only process one fraction at a time.

lapply(xft, frac_to_numeric)
# Please note that this is the correct way to work with several fractions at once.




# Example 5

library(iemisc)

pix <- "270/11"

pi1 <- "22/7" # Reference 3

pi2 <- "355/113" # Reference 3

frac_to_numeric(pix)

frac_to_numeric(pi1)

frac_to_numeric(pi2)




# Example 6

# If you have a construction measurement that includes a dimension in feet,
# such as 49 ft 7 5/8 in, don't use the frac_to_numeric function, instead
# use the construction_fraction function.

library(iemisc)

xxift <- "49 ft 7 5/8 in"

construction_decimal(xxift, result = "traditional", output = "vector")




# Example 7 -- Reference 2

truss_marks <- "19 3/16 inches"
```

```
frac_to_numeric(truss_marks)
```

---

igivenICPn *Simple Interest rate given Interest Charged, Number of years, and Principal value*

Description

Compute i given IC, n, and P

Usage

```
igivenICPn(P, IC, n = NULL, begin_event = NULL, end_event = NULL)
```

Arguments

P	numeric vector that contains the principal value
IC	numeric vector that contains the total amount of interest charged
n	numeric vector that contains the number of years
begin_event	character vector that contains the start date
end_event	character vector that contains the end date

Details

i is expressed as

$$i = \frac{IC * 100}{P * n}$$

i the interest rate

IC the total amount of interest charged

P the principal amount of the loan

n the number of years

Value

i numeric vector that contains the simple interest rate as a percent rounded to 2 decimal places

Author(s)

Irucka Embry

References

TN Code § 47-14-102 (2021). 2021 Tennessee Code – Title 47 - Commercial Instruments and Transactions – Chapter 14 - Interest Rates Generally – § 47-14-102. Definitions. See https://law.justia.com/codes/tennessee/2021/title-47/chapter-14/section-47-14-102/.

Examples

```
# Example 1

library(iemisc)

igivenICPn(P = 500, IC = 1000, n = 10)



# Example 2

library(iemisc)

igivenICPn(P = 500, IC = 1000, begin_event = "1 January 2020", end_event = "1 January 2030")
```

---

igivenPFn — *Interest rate given Future value, Number of periods, and Present value (Engineering Economics)*

Description

Compute i given F, n, and P

Usage

```
igivenPFn(P, F, n)
```

Arguments

P	numeric vector that contains the present value(s)
F	numeric vector that contains the future value(s)
n	numeric vector that contains the period value(s)

Details

i is expressed as

$$i = \sqrt[n]{\frac{F}{P}} - 1$$

i the "effective interest rate per interest period"

F the "future equivalent"

P the "present equivalent"

n the "number of interest periods"

Value

i numeric vector that contains the effective interest rate as a percent rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 128-129, 142.

Examples

```
# Example for equation 4-6 from the Reference text (page 128)

library(iemisc)

igivenPFn(P = 500, F = 1000, n = 10)
```

`iscolumn` *Column Vector (GNU Octave/MATLAB compatible)*

Description

Test for column vector that is compatible with GNU Octave/MATLAB.

Usage

```
iscolumn(x)
```

Arguments

`x`	An array (array, matrix, vector)

Value

"Return true if x is a column vector. A column vector is a 2-D array for which size (x) returns [N, 1] with non-negative N." Source: Eaton page 68.

Author(s)

Irucka Embry, Rik Wehbring (GNU Octave), Colin B. Macdonald (OctSymPy)

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 68.

See Also

isrow

Examples

```
library(iemisc)

# Examples

xxx <- ramify::mat("1, 2"); xxx

iscolumn(xxx)
```

---

isrow *Row Vector (GNU Octave/MATLAB compatible)*

Description

Test for row vector that is compatible with GNU Octave/MATLAB.

Usage

```
isrow(x)
```

Arguments

x	An array (array, matrix, vector)

Value

"Return true if x is a row vector. A row vector is a 2-D array for which size (x) returns [1, N] with non-negative N." Source: Eaton page 68.

Author(s)

Irucka Embry, Rik Wehbring (GNU Octave), Colin B. Macdonald (OctSymPy)

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 68.

See Also

iscolumn

Examples

```
library(iemisc)

# Examples

xx <- ramify::mat("1, 2"); xx

isrow(xx)
```

---

kin_visc_water *Kinematic Viscosity for liquid Water*

Description

This function solves for the kinematic viscosity of water using only the water density and the dynamic viscosity.

Usage

```
kin_visc_water(
  rho,
  mu,
  rho_units = c("kg/m^3", "lbm/ft^3", "slug/ft^3"),
  mu_units = c("Pa*s or kg/m/s", "lbf*s/ft^2", "slug/ft/s")
)
```

Arguments

`rho`	numeric vector that contains the water density
`mu`	numeric vector that contains the water dynamic viscosity
`rho_units`	character vector that contains the unit for the density of water [options are kg/m^3, lbm/ft^3, or slug/ft^3]
`mu_units`	character vector that contains the unit for the dynamic viscosity of water [options are Pa*s or kg/m/s, lbf*s/ft^2, or slug/ft/s]

Details

The simplified equation is expressed as

$$nu = \frac{mu}{rho}$$

\nu Water Kinematic Viscosity (m^2/s or ft^2/s)

\rho Water Density (mass divided by volume), slug/ft^3

\mu Water Dynamic viscosity, slug/ft/s

Value

the kinematic viscosity as a numeric vector. The units are not returned.

Note

Note: 1 lbf = 1 slug * 1 ft/sec^2, thus 1 slug = 1 lbf * sec^2 / 1 ft (Reference 2)

Thus, lbm/ft^3 = lbf*s^2/ft/ft^3

Author(s)

Irucka Embry

Source

1. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See `https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2`.
2. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See `https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-aske`

References

1. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 7-8.
2. Professor S.A. Kinnas, Commonly used units in CE319F (Elementary Fluid Mechanics), The University of Texas at Austin Department of Civil, Architectural and Environmental Engineering, `https://www.caee.utexas.edu/prof/kinnas/319LAB/notes13/units_ce319f_kinnas.pdf`.

Examples

```
# Example 1


# See Source 1 and Source 2

library(iemisc)

try(kin_visc_water(mu = 34, rho = 0, rho_units = "kg/m^3", mu_units = "Pa*s or kg/m/s"))





# Example 2 (from the Reference)

install.load::load_package("iemisc", "units")

import::from(fpCompare, "%==%")


# For water at 68 F (20 C), mu = 2.09 * 10 ^ -8 slug/ft/s and rho = 1.937 slug/ft^3

kin_visc_water(mu = 2.09 * 10 ^ -8, rho = 1.937, rho_units =
"slug/ft^3", mu_units = "slug/ft/s")

# convert the units

rho <- set_units(1.937, slug/ft^3)

mu <- set_units(2.09 * 10 ^ -8, slug/ft/s)

mu1 <- set_units(mu, kg/m/s)

rho1 <- set_units(rho, "kg/m^3")

kin_visc_water(mu = mu1, rho = rho1, rho_units = "kg/m^3", mu_units =
"Pa*s or kg/m/s")

mu2 <- set_units(mu, lbf*s/ft^2)

rho2 <- set_units(rho, lb/ft^3)

kin_visc_water(mu = mu2, rho = rho2, rho_units = "lbm/ft^3", mu_units =
"lbf*s/ft^2")


# compare the results of part 1 and part 3 (they should be equivalent)

kin_visc_water(mu = 2.09 * 10 ^ -8, rho = 1.937, rho_units = "slug/ft^3",
mu_units = "slug/ft/s") %==% kin_visc_water(mu = mu2, rho = rho2, rho_units =
"lbm/ft^3", mu_units = "lbf*s/ft^2")
```

```
# Example 2 - Example from the hydraulics package

install.load::load_package("iemisc", "units")

import::from(fpCompare, "%==%")


nu <- hydraulics::kvisc(T = 55, units = "Eng", ret_units = TRUE); nu

nus <- hydraulics::dvisc(T = 55, units = "Eng", ret_units = TRUE) /
hydraulics::dens(T = 55, units = "Eng", ret_units = TRUE); nus


# compare the results of nu and nus (they should be equivalent)

drop_units(nu) %==% drop_units(nus)


nu2 <- dyn_visc_water(Temp = 55, units = "Eng", Eng_units = "lbf*s/ft^2") /
density_water(Temp = 55, units = "Eng", Eng_units = "slug/ft^3"); nu2

nus2 <- kin_visc_water(mu = dyn_visc_water(Temp = 55, units = "Eng", Eng_units =
"lbf*s/ft^2"), rho = density_water(Temp = 55, units = "Eng", Eng_units = "slug/ft^3"),
rho_units = "lbm/ft^3", mu_units = "lbf*s/ft^2"); nus2

# compare the results of nu2 and nus2 (they should be equivalent)

nu2 %==% nus2
```

`lat_long2state` *United States of America (USA) State Identification Using Latitude/Longitude Coordinates*

Description

Using the provided latitude/longitude coordinates (as character or numeric vectors), this function determines whether the location is within an United States of America (USA) state/commonwealth, Puerto Rico, or the U.S. Virgin Islands

Usage

```
lat_long2state(latitude, longitude)
```

Arguments

`latitude`	numeric vector (or character vector with numbers only) that contains the latitude as a decimal degree
`longitude`	numeric vector (or character vector with numbers only) that contains the longitude as a decimal degree

Value

the location name as a character vector (United States of America (USA) state/commonwealth, Puerto Rico, or the U.S. Virgin Islands

Author(s)

Irucka Embry, Josh O'Brien (Stack Overflow R code)

Source

Latitude Longitude Coordinates to State Code in R - Stack Overflow answered by Josh O'Brien on Jan 6 2012 and edited by Josh O'Brien on Jun 18, 2020. See https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r.

Examples

```
# Example 1

library(iemisc)

lat_long2state(latitude = c(36.3684553, 40), longitude = c(-82.1796880, -89))
lat_long2state(latitude = "36.3684553", longitude = "-82.1796880")



# Example 2

# Test the function using points in Wisconsin and Oregon (From Source 1)

library(iemisc)

x = c(-90, -120); y = c(44, 44)
lat_long2state(latitude = y, longitude = x)
```

`lat_long2utm` *Conversion of Latitude/Longitude to UTM Coordinates*

Description

Takes latitude/longitude coordinates (as character or numeric vectors) and transforms them into their respective UTM Easting and Northing coordinates (with units of US Survey foot, foot, or meters) & UTM Zone.

Usage

```
lat_long2utm(
  latitude,
  longitude,
  units = c("us-ft", "ft", "m"),
  output = c("basic", "table")
)
```

Arguments

`latitude`	numeric vector (or character vector with numbers only) that contains the latitude as a decimal degree
`longitude`	numeric vector (or character vector with numbers only) that contains the longitude as a decimal degree
`units`	character vector that contains the system of units (options are `survey_ft` (United States Customary System) [US survey foot], `foot`, or `meters` (International System of Units) [meters])
`output`	character vector that contains basic for the default result using a `list` or table for the result as a `data.table`

Value

the UTM zone along with the UTM Easting and Northing coordinates (in the requested unit) as either a list or a data.table

Note

Remember: Latitude coordinates signify North (N) or South (S) while longitude coordinates signify East (E) and West (W). It is customary to denote West longitude coordinates and South latitude coordinates as negative (-).

Stack Overflow user contributions are "licensed under CC BY-SA 3.0 with attribution required." [Stack Overflow Reference] I have decided to make my adaptions to the Stack Overflow user contributions as CC BY-SA 4.0 thereby enabling me to license my adaptions to the aforementioned code as GPLv3. [Creative Commons References]

Author(s)

Irucka Embry, Teodor Ciuraru (Latitude/Longitude to UTM conversion code), and Josh O'Brien (Latitude/Longitude to UTM conversion code)

Source

1. r - Converting latitude and longitude points to UTM - Stack Overflow answered and edited by Teodor Ciuraru on Feb 17 2018. See https://stakoverflow.com/questions/18639967/converting-latitude-and-longitude-points-to-utm.
2. r - Converting latitude and longitude points to UTM - Stack Overflow answered by Josh O'Brien on Sep 5 2013 and edited by Josh O'Brien on Feb 21 2014. See https://stakoverflow.com/questions/18639967/converting-latitude-and-longitude-points-to-utm.
3. r - data.table alternative for dplyr mutate? - Stack Overflow answered by Arun on Aug 30 2015. See https://stackoverflow.com/questions/29583665/data-table-alternative-for-dplyr-mutate.
4. database design - What is the maximum length of latitude and longitude? - Stack Overflow answered by JasonM1 on May 24 2013 and edited by JasonM1 on Jul 16 2019. See https://stackoverflow.com/questions/15965166/what-is-the-maximum-length-of-latitude-and-longitude.
5. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
6. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-askec
7. Latitude Longitude Coordinates to State Code in R - Stack Overflow answered by Josh O'Brien on Jan 6 2012 and edited by Josh O'Brien on Jun 18, 2020. See https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r.

References

1. MapTools, 29 May 2016, "More details about UTM grid zones", https://www.maptools.com/tutorials/grid_zone_details.
2. Wikimedia Foundation, Inc. Wikipedia, 11 August 2019, "Geographic coordinate system", https://en.wikipedia.org/wiki/Geographic_coordinate_system.
3. PROJ 6.2.0 documentation, 28 Oct 2019, "Cartographic projection", https://proj.org/usage/projections.html.
4. Wikimedia Foundation, Inc. Wikibooks, 19 August 2018, "PROJ.4", https://en.wikibooks.org/wiki/PROJ.4.
5. National Geospatial-Intelligence Agency Office of Geomatics, "Military Grid Reference System (MGRS) Grid Zone Designator (GZD's)", https://vdocuments.net/military-grid-reference-system-mgrs html.
6. Tennessee Department of Transportation Design Division, Tennessee Department of Transportation Tennessee Geodetic Reference Network (TGRN) Reference Manual Second Edition Issued, page ix, https://www.tn.gov/content/dam/tn/tdot/documents/TgrnComposite.pdf.

7. LatLong.net, "Lat Long to UTM Converter", https://www.latlong.net/lat-long-utm.html.
8. NOAA's National Geodetic Survey (NGS), "NGS Coordinate Conversion and Transformation Tool (NCAT)", https://www.ngs.noaa.gov/NCAT/.
9. Creative Commons. Weblog Archives, October 8, 2015, "CC BY-SA 4.0 now one-way compatible with GPLv3: The declaration increases interoperability of the commons for games, hardware designs, and more" Posted by mike, https://creativecommons.org/2015/10/08/cc-by-sa-4-0-now-one-way-compatible-with-gplv3/.
10. Stack Overflow. Public Network Terms of Service, "6. Content Permissions, Restrictions, and Creative Commons Licensing", https://stackoverflow.com/legal/terms-of-service#licensing.

Examples

```
# Example 1
# Test location from TGRN Reference Manual with NCAT
# using the 1983 (1995) DATUM
# GPS 1 is the station name with these coordinates
# latitude (North) = 36 22 6.43923
# longitude (West) = 82 10 46.87679

install.load::load_package("iemisc", "sp")

lats <- as.numeric(char2dms("36d22'6.43923\"N"))
lats

longs <- as.numeric(char2dms("82d10'46.87679\"W"))
longs

latsc <- as.character(lats)
latsc

longsc <- as.character(longs)
longsc

lat_long2utm(latsc, longsc, units = "m", output = "basic")

lat_long2utm(latsc, longsc, units = "m", output = "table")

lat_long2utm(lats, longs, units = "m", output = "basic")

lat_long2utm(lats, longs, units = "m", output = "table")

# From https://www.ngs.noaa.gov/NCAT/
# Latitude: 36.3684553416667
# Longitude: -82.1796879972222
# UTM Northing (m): 4,025,462.877
# UTM Easting (m): 394,172.067
# USNG: 17SLA9417225462
```

```
# Example 2
# Test against Grid [Reference: National Geospatial-Intelligence Agency Office of Geomatics]

library(iemisc)

lat_long2utm("80", "-179", units = "m", output = "basic") # = 1X

lat_long2utm("-80", "-179", units = "m", output = "basic") # = 1C



# Example 3
# Test with world cities


# See Source 5 and Source 6

install.load::load_package("iemisc", "maps", "rando", "utils", "data.table")

import::from(sampler, rsamp)

data(world.cities) # from maps

set_n(200) # makes the example reproducible

wc <- rsamp(world.cities, 2, over = 0, rep = FALSE)
wc

wcutm1 <- lat_long2utm(wc$lat[1], wc$long[1], units = "m", output = "table")
wcutm1

wcutm2 <- lat_long2utm(wc$lat[2], wc$long[2], units = "m", output = "table")
wcutm2

l <- list(wcutm1, wcutm2)
ll <- rbindlist(l)

wc_utm <- setDT(cbind(wc, ll))
wc_utm




# Example 4
# Test with 2 Web sites

library(iemisc)

latlong1 <- lat_long2utm(6.32, 7.41, units = "m", output = "table")
latlong1

latlong2 <- lat_long2utm(44.47, 19.81, units = "m", output = "table")
```

```
latlong2


# Results from https://www.latlong.net/lat-long-utm.html
# Latitude: 6.32
# Longitude: 7.41
# UTM Easting: 324118.76
# UTM Northing: 698846.97
# UTM Zone: 32N

# Latitude: 44.47
# Longitude: 19.81
# UTM Easting: 405349.04
# UTM Northing: 4924765.48
# UTM Zone: 34T


# Results from https://www.ngs.noaa.gov/NCAT/
# Latitude: 6.32
# Longitude: 7.41
# UTM Northing (m): 698,846.969
# UTM Easting (m): 324,118.758
# USNG: 32NLM2411898846

# Latitude: 44.47
# Longitude: 19.81
# UTM Northing (m): 4,924,765.484
# UTM Easting (m): 405,349.043
# USNG: 34TDQ0534924765
```

`length_octave` *Length of R objects (GNU Octave/MATLAB compatible)*

Description

Obtain the length of R objects [arrays, matrices, and vectors (including lists)] in a manner compatible with GNU Octave/MATLAB. Some documentation from `length`.

Usage

```
length_octave(x)
```

Arguments

x An R object (array, matrix, vector)

Value

Return the length of the object x as an integer. "The length is 0 for empty objects, 1 for scalars (in R, a `vector` of `length` 1), and the number of elements (in R, the `length`) for vectors. For matrix objects, the length is the number of rows or columns, whichever is greater (this odd definition is used for compatibility with MATLAB)." Source: Eaton.

Author(s)

Irucka Embry, Samit Basu (FreeMat)

References

1. Samit Basu (2002-2006). FreeMat v4.0, https://freemat.sourceforge.net/help/inspection_length.html.
2. John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 47.

See Also

length, lengths, size, size

Examples

```
library(iemisc)

import::from(matlab, ones)

# Example from pracma isempty

object1 <- matrix(0, 1, 0)

length_octave(object1)
```

`lookupQT` *Hash Table/Dictionary Lookup These functions were originally contained in "qdapTools" version 1.3.4* `lookup` - R*hrefhttp://datatable.r-forge.r-project.org/***data.table** *based hash table useful for large vector lookups.*

Description

Hash Table/Dictionary Lookup

These functions were originally contained in "qdapTools" version 1.3.4 `lookup` - **data.table** based hash table useful for large vector lookups.

Usage

```
lookupQT(terms, key.match, key.reassign = NULL, missing = NA)
```

Arguments

`terms`	A vector of terms to undergo a lookup.
`key.match`	Takes one of the following: (1) a two column data.frame of a match key and reassignment column, (2) a named list of vectors (Note: if data.frame or named list supplied no key reassign needed) or (3) a single vector match key.
`key.reassign`	A single reassignment vector supplied if key.match is not a two column data.frame/named list.
`missing`	Value to assign to terms not matching the key.match. If set to `NULL` the original values in `terms` corresponding to the missing elements are retained.

Value

Outputs A new vector with reassigned values.

Author(s)

Tyler Rinker ('qdapTools' package version 1.3.4)

`Manningcirc` *Circular Cross-section Using the Gauckler-Manning-Strickler Equation 1*

Description

Manningcirc and Manningcircy solve for a missing variable for a circular cross-section. The `uniroot` function is used to obtain the missing parameters.

The Manningcirc function solves for one missing variable in the Gauckler- Manning equation for a circular cross-section and uniform flow. The possible inputs are Q, n, Sf, y, and d. If y or d are not initially known, then Manningcircy can solve for y or d to use as input in the Manningcirc function.

Usage

```
Manningcirc(
  Q = NULL,
  n = NULL,
  Sf = NULL,
  y = NULL,
  d = NULL,
  Temp = NULL,
  units = c("SI", "Eng")
)
```

Arguments

`Q`	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
`n`	numeric vector that contains the Manning's roughness coefficient n, if known.
`Sf`	numeric vector that contains the bed slope (m/m or ft/ft), if known.
`y`	numeric vector that contains the flow depth (m or ft), if known.
`d`	numeric vector that contains the diameters value (m or ft), if known.
`Temp`	numeric vector that contains the temperature (degrees C or degrees Fahrenheit), if known.
`units`	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]

Details

Circular cross-section using the Gauckler-Manning-Strickler equation

Gauckler-Manning-Strickler equation is expressed as

$$V = \frac{K_n}{n} R^{\frac{2}{3}} \sqrt{S}$$

V the velocity (m/s or ft/s)

n Manning's roughness coefficient (dimensionless)

R the hydraulic radius (m or ft)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

This equation is also expressed as

$$Q = \frac{K_n}{n} \frac{A^{\frac{5}{3}}}{P^{\frac{2}{3}}} \sqrt{S}$$

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

n Manning's roughness coefficient (dimensionless)

P the wetted perimeters of the channel (m or ft)

A the cross-sectional area (m^2 or ft^2)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

Other important equations regarding the circular cross-section follow:

$$R = \frac{A}{P}$$

R the hydraulic radius (m or ft)

A the cross-sectional area (m^2 or ft^2)

P the wetted perimeters of the channel (m or ft)

$$A = (theta - \sin theta)\,\frac{d^2}{8}$$

A the cross-sectional area (m^2 or ft^2)

d the diameters of the cross-section (m or ft)

\theta see the equation defining this parameters

$$theta = 2 \arcsin \left[1 - 2\left(\frac{y}{d}\right)\right]$$

\theta see the equation defining this parameters

y the flow depth (normal depth in this function) [m or ft]

d the diameters of the cross-section (m or ft)

$$d = 1.56 \left[\frac{nQ}{K_n \sqrt{S}}\right]^{\frac{3}{8}}$$

d the initial diameters of the cross-section [m or ft]

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

n Manning's roughness coefficient (dimensionless)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

Note: This will only provide the initial conduit diameters, check the design considerations to determine your next steps.

$$P = \frac{thetad}{2}$$

P the wetted perimeters of the channel (m or ft)

\theta see the equation defining this parameters

d the diameters of the cross-section (m or ft)

$$B = d\sin\left(\frac{theta}{2}\right)$$

B the top width of the channel (m or ft)

\theta see the equation defining this parameters

d the diameters of the cross-section (m or ft)

$$D = \frac{A}{B}$$

D the hydraulic depth (m or ft)

A the cross-sectional area (m^2 or ft^2)

B the top width of the channel (m or ft)

$$Z = \frac{\sqrt{2}}{2}my^2.5$$

Z the Section factor (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$E = y + \frac{Q^2}{2gA^2}$$

E the Specific Energy (m or ft)

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

g gravitational acceleration (m/s^2 or ft/sec^2)

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

$$VH = \frac{V^2}{2g}$$

VH the Velocity Head (m or ft)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

A rough turbulent zone check is performed on the water flowing in the channel using the Reynolds number (Re). The Re equation follows:

$$Re = \frac{rhoRV}{mu}$$

Re Reynolds number (dimensionless)

\rho density (kg/m^3 or slug/ft^3)

R the hydraulic radius (m or ft)

V the velocity (m/s or ft/s)

\mu dynamic viscosity (* 10^-3 kg/m*s or * 10^-5 lb*s/ft^2)

A critical flow check is performed on the water flowing in the channel using the Froude number (Fr). The Fr equation follows:

$$Fr = \frac{V}{\left(\sqrt{g * D}\right)}$$

Fr the Froude number (dimensionless)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

D the hydraulic depth (m or ft)

Value

the missing parameters (Q, n, or Sf) & theta, area (A), wetted perimeters (P), velocity (V), top width (B), hydraulic depth (D), hydraulic radius (R), E (Specific Energy), Vel_Head (Velocity Head), Z (Section Factor), Reynolds number (Re), and Froude number (Fr) as a `list`. for the Manningcirc function.

Note

Assumptions: uniform flow, prismatic channel, and surface water temperature of 20 degrees Celsius (68 degrees Fahrenheit) at atmospheric pressure

Note: Units must be consistent

Please refer to the iemisc: Manning... Examples using iemiscdata [https://www.ecoccs.com/R_Examples/Manning_iemiscdat and iemisc: Open Channel Flow Examples involving Geometric Shapes with the Gauckler-Manning-Strickler Equation [https://www.ecoccs.com/R_Examples/Open-Channel-Flow_Examples_Geometric_Shapes.pdf] for the cross-section examples using iemiscdata

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 2, 8, 36, 102, 120, 123-125, 153-154.
2. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
3. Gilberto E. Urroz, Utah State University Civil and Environmental Engineering - OCW, CEE6510 - Numerical Methods in Civil Engineering, Spring 2006 (2006). Course 3. "Solving selected equations and systems of equations in hydraulics using Matlab", August/September 2004, https://digitalcommons.usu.edu/ocw_cee/3/.
4. Tyler G. Hicks, P.E., *Civil Engineering Formulas: Pocket Guide*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2002, page 423, 425.
5. Wikimedia Foundation, Inc. Wikipedia, 26 November 2015, "Manning formula", https://en.wikipedia.org/wiki/Manning_formula.
6. John C. Crittenden, R. Rhodes Trussell, David W. Hand, Kerry J. Howe, George Tchobanoglous, *MWH's Water Treatment: Principles and Design*, Third Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2012, page 1861-1862.
7. Andrew Chadwick, John Morfett and Martin Borthwick, *Hydraulics in Civil and Environmental Engineering*, Fourth Edition, New York City, New York: Spon Press, Inc., 2004, page 133.
8. Robert L. Mott and Joseph A. Untener, *Applied Fluid Mechanics*, Seventh Edition, New York City, New York: Pearson, 2015, page 376, 377-378, 392.
9. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 21, 40-41.
10. Gary P. Merkley, "BIE6300 - Irrigation & Conveyance Control Systems, Spring 2004", 2004, Biological and Irrigation Engineering - OCW. Course 2, https://digitalcommons.usu.edu/ocw_bie/2/.
11. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://physics.nist.gov/cgi-bin/cuu/Value?gn.
12. Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, "Conversion of units", https://en.wikipedia.org/wiki/Conversion_of_units.

See Also

`Manningtrap` for a trapezoidal cross-section, `Manningrect` for a rectangular cross-section, `Manningtri` for a triangular cross-section, and `Manningpara` for a parabolic cross-section.

`Manningcircy`

`Manningcircy` *Circular Cross-section Using the Gauckler-Manning-Strickler Equation 2*

Description

The Manningcircy function solves for one missing variable in the Gauckler- Manning equation for a circular cross-section and uniform flow. The possible inputs are y, d, y_d (ratio of y/d), and theta.

Usage

```
Manningcircy(
  y = NULL,
  d = NULL,
  y_d = NULL,
  theta = NULL,
  Sf = NULL,
  Q = NULL,
  units = c("SI", "Eng")
)
```

Arguments

`y`	numeric vector that contains the flow depth (m or ft), if known.
`d`	numeric vector that contains the diameters value (m or ft), if known.
`y_d`	numeric vector that contains the filling ration (y/d), if known.
`theta`	numeric vector that contains the angle theta (radians), if known.
`Sf`	numeric vector that contains the bed slope (m/m or ft/ft), if known.
`Q`	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
`units`	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]

Value

the missing parameters (d or y) & theta, area (A), wetted perimeters (P), top width (B), velocity (V), hydraulic depth (D), hydraulic radius (R), E (Specific Energy), Vel_Head (Velocity Head), Z (Section Factor), Reynolds number (Re), and Froude number (Fr) as a `list`. for the Manningcircy function.

Author(s)

Irucka Embry

See Also

`Manningcirc` for the examples section

Manningpara *Parabolic cross-section for the Gauckler-Manning-Strickler equation*

Description

This function solves for one missing variable in the Gauckler-Manning- Strickler equation for a parabolic cross-section and uniform flow. The uniroot function is used to obtain the missing parameters.

Usage

```
Manningpara(
  Q = NULL,
  n = NULL,
  m = NULL,
  Sf = NULL,
  y = NULL,
  B1 = NULL,
  y1 = NULL,
  Temp = NULL,
  units = c("SI", "Eng")
)
```

Arguments

Q	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
n	numeric vector that contains the Manning's roughness coefficient n, if known.
m	numeric vector that contains the "cross-sectional side slope of m:1 (horizontal:vertical)", if known.
Sf	numeric vector that contains the bed slope (m/m or ft/ft), if known.
y	numeric vector that contains the flow depth (m or ft), if known.
B1	numeric vector that contains the "bank-full width", if known.
y1	numeric vector that contains the "bank-full depth", if known.
Temp	numeric vector that contains the temperature (degrees C or degrees Fahrenheit), if known.
units	character vector that contains the system of units [options are SI for International System of Units or Eng for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]

Details

Gauckler-Manning-Strickler equation is expressed as

$$V = \frac{K_n}{n} R^{\frac{2}{3}} S^{\frac{1}{2}}$$

V the velocity (m/s or ft/s)

n Manning's roughness coefficient (dimensionless)

R the hydraulic radius (m or ft)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

This equation is also expressed as

$$Q = \frac{K_n}{n} \frac{A^{\frac{5}{3}}}{P^{\frac{2}{3}}} S^{\frac{1}{2}}$$

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

n Manning's roughness coefficient (dimensionless)

P the wetted perimeters of the channel (m or ft)

A the cross-sectional area (m^2 or ft^2)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

Other important equations regarding the parabolic cross-section follow:

$$R = \frac{A}{P}$$

R the hydraulic radius (m or ft)

A the cross-sectional area (m^2 or ft^2)

P the wetted perimeters of the channel (m or ft)

$$A = \frac{2}{3} B y$$

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

B the top width of the channel (m or ft)

$$P = \left(\frac{B}{2}\right) \left[\sqrt{(1 + x^2)} + \left(\frac{1}{x}\right) \ln\left(x + \sqrt{(1 + x^2)}\right)\right]$$

P the wetted perimeters of the channel (m or ft)

B the top width of the channel (m or ft)

x 4y/b (dimensionless)

$$x = \frac{4y}{B}$$

x 4y/b (dimensionless)

B the top width of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

$$B = B_1 \left(\sqrt{\frac{y}{y_1}} \right)$$

B the top width of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

B_1 the "bank-full width" (m or ft)

y_1 the "bank-full depth" (m or ft)

$$D = \frac{A}{B}$$

D the hydraulic depth (m or ft)

A the cross-sectional area (m^2 or ft^2)

B the top width of the channel (m or ft)

$$Z = \frac{\sqrt{2}}{2} m y^2 .5$$

Z the Section factor (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$E = y + \frac{Q^2}{2gA^2}$$

E the Specific Energy (m or ft)

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

g gravitational acceleration (m/s^2 or ft/sec^2)

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

$$VH = \frac{V^2}{2g}$$

VH the Velocity Head (m or ft)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

A rough turbulent zone check is performed on the water flowing in the channel using the Reynolds number (Re). The Re equation follows:

$$Re = \frac{rhoRV}{mu}$$

Re Reynolds number (dimensionless)

\rho density (kg/m^3 or slug/ft^3)

R the hydraulic radius (m or ft)

V the velocity (m/s or ft/s)

\mu dynamic viscosity (* 10^-3 kg/m*s or * 10^-5 lb*s/ft^2)

A critical flow check is performed on the water flowing in the channel using the Froude number (Fr). The Fr equation follows:

$$Fr = \frac{V}{(\sqrt{g * D})}$$

Fr the Froude number (dimensionless)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

D the hydraulic depth (m or ft)

Value

the missing parameter (Q, n, m, Sf, B1, y1, or y) & area (A), wetted perimeter (P), velocity (V), top width (B), hydraulic radius (R), Reynolds number (Re), and Froude number (Fr) as a `list`.

Note

Assumptions: uniform flow, prismatic channel, and surface water temperature of 20 degrees Celsius (68 degrees Fahrenheit) at atmospheric pressure

Note: Units must be consistent

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 2, 8, 36, 102, 120, 153.
2. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

3. Gilberto E. Urroz, Utah State University Civil and Environmental Engineering - OCW, CEE6510 - Numerical Methods in Civil Engineering, Spring 2006 (2006). Course 3. "Solving selected equations and systems of equations in hydraulics using Matlab", August/September 2004, https://digitalcommons.usu.edu/ocw_cee/3/.
4. Tyler G. Hicks, P.E., *Civil Engineering Formulas: Pocket Guide*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2002, page 423, 425.
5. Wikimedia Foundation, Inc. Wikipedia, 26 November 2015, "Manning formula", https://en.wikipedia.org/wiki/Manning_formula.
6. John C. Crittenden, R. Rhodes Trussell, David W. Hand, Kerry J. Howe, George Tchobanoglous, *MWH's Water Treatment: Principles and Design*, Third Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2012, page 1861-1862.
7. Andrew Chadwick, John Morfett and Martin Borthwick, *Hydraulics in Civil and Environmental Engineering*, Fourth Edition, New York City, New York: Spon Press, Inc., 2004, page 133.
8. Robert L. Mott and Joseph A. Untener, *Applied Fluid Mechanics*, Seventh Edition, New York City, New York: Pearson, 2015, page 376.
9. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 21, 40-41.
10. Gary P. Merkley, "BIE6300 - Irrigation & Conveyance Control Systems, Spring 2004", 2004, Biological and Irrigation Engineering - OCW. Course 2, https://digitalcommons.usu.edu/ocw_bie/2/.
11. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://physics.nist.gov/cgi-bin/cuu/Value?gn.
12. Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, "Conversion of units", https://en.wikipedia.org/wiki/Conversion_of_units.

See Also

Manningtrap for a trapezoidal cross-section, Manningrect for a rectangular cross-section, Manningtri for a triangular cross-section, and Manningcirc for a circular cross-section.

Examples

```
library(iemisc)

# Exercise 4.3 from Sturm (page 153)
y <- Manningpara(Q = 12.0, B1 = 10, y1 = 2.0, Sf = 0.005, n = 0.05, units = "SI")
# defines all list values within the object named y
# Q = 12.0 m^3/s, B1 = 10 m, y1 = 2.0 m, Sf = 0.005 m/m, n = 0.05, units = SI units
# This will solve for y since it is missing and y will be in m

y$y # gives the value of y



# Modified Exercise 4.3 from Sturm (page 153)
```

```
Manningpara(y = y$y, B1 = 10, y1 = 2.0, Sf = 0.005, n = 0.05, units = "SI")
# y = 1.254427 m, B1 = 10 m, y1 = 2.0 m, Sf = 0.005 m/m, n = 0.05, units = SI units
# This will solve for Q since it is missing and Q will be in m^3/s
```

`Manningrect` *Rectangular cross-section for the Gauckler-Manning-Strickler equation*

Description

This function solves for one missing variable in the Gauckler-Manning- Strickler equation for a rectangular cross-section and uniform flow. The `uniroot` function is used to obtain the missing parameters.

Usage

```
Manningrect(
  Q = NULL,
  n = NULL,
  b = NULL,
  Sf = NULL,
  y = NULL,
  Temp = NULL,
  units = c("SI", "Eng")
)
```

Arguments

`Q`	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
`n`	numeric vector that contains the Manning's roughness coefficient n, if known.
`b`	numeric vector that contains the bottom width, if known.
`Sf`	numeric vector that contains the bed slope (m/m or ft/ft), if known.
`y`	numeric vector that contains the flow depth (m or ft), if known.
`Temp`	numeric vector that contains the temperature (degrees C or degrees Fahrenheit), if known.
`units`	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]

Details

Gauckler-Manning-Strickler equation is expressed as

$$V = \frac{K_n}{n} R^{\frac{2}{3}} S^{\frac{1}{2}}$$

V the velocity (m/s or ft/s)

n Manning's roughness coefficient (dimensionless)

R the hydraulic radius (m or ft)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

This equation is also expressed as

$$Q = \frac{K_n}{n} \frac{A^{\frac{5}{3}}}{P^{\frac{2}{3}}} S^{\frac{1}{2}}$$

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

n Manning's roughness coefficient (dimensionless)

P the wetted perimeters of the channel (m or ft)

A the cross-sectional area (m^2 or ft^2)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

Other important equations regarding the rectangular cross-section follow:

$$R = \frac{A}{P}$$

R the hydraulic radius (m or ft)

A the cross-sectional area (m^2 or ft^2)

P the wetted perimeters of the channel (m or ft)

$$A = by$$

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

b the bottom width (m or ft)

$$P = b + 2y$$

P the wetted perimeters of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

b the bottom width (m or ft)

$$B = b$$

B the top width of the channel (m or ft)

b the bottom width (m or ft)

$$D = \frac{A}{B}$$

D the hydraulic depth (m or ft)

A the cross-sectional area (m^2 or ft^2)

B the top width of the channel (m or ft)

$$Z = \frac{\sqrt{2}}{2} m y^2 .5$$

Z the Section factor (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$E = y + \frac{Q^2}{2gA^2}$$

E the Specific Energy (m or ft)

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

g gravitational acceleration (m/s^2 or ft/sec^2)

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

$$VH = \frac{V^2}{2g}$$

VH the Velocity Head (m or ft)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

A rough turbulent zone check is performed on the water flowing in the channel using the Reynolds number (Re). The Re equation follows:

$$Re = \frac{rhoRV}{mu}$$

Re Reynolds number (dimensionless)

\rho density (kg/m^3 or slug/ft^3)

R the hydraulic radius (m or ft)

V the velocity (m/s or ft/s)

\mu dynamic viscosity (* 10^-3 kg/m*s or * 10^-5 lb*s/ft^2)

A critical flow check is performed on the water flowing in the channel using the Froude number (Fr). The Fr equation follows:

$$Fr = \frac{V}{(\sqrt{g * D})}$$

Fr the Froude number (dimensionless)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

D the hydraulic depth (m or ft)

Value

the missing parameters (Q, n, b, Sf, or y) & area (A), wetted perimeter (P), velocity (V), top width (B), hydraulic radius (R), Reynolds number (Re), and Froude number (Fr) as a `list`.

Note

Assumptions: uniform flow, prismatic channel, and surface water temperature of 20 degrees Celsius (68 degrees Fahrenheit) at atmospheric pressure

Note: Units must be consistent

Please refer to the iemisc: Manning... Examples using iemiscdata [https://www.ecoccs.com/R_Examples/Manning_iemiscdat and iemisc: Open Channel Flow Examples involving Geometric Shapes with the Gauckler-Manning-Strickler Equation [https://www.ecoccs.com/R_Examples/Open-Channel-Flow_Examples_Geometric_Shapes.pdf] for the cross-section examples using iemiscdata

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 2, 8, 36, 102, 120, 153-154.
2. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

3. Gilberto E. Urroz, Utah State University Civil and Environmental Engineering - OCW, CEE6510 - Numerical Methods in Civil Engineering, Spring 2006 (2006). Course 3. "Solving selected equations and systems of equations in hydraulics using Matlab", August/September 2004, https://digitalcommons.usu.edu/ocw_cee/3/.

4. Tyler G. Hicks, P.E., *Civil Engineering Formulas: Pocket Guide*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2002, page 423, 425.

5. Wikimedia Foundation, Inc. Wikipedia, 26 November 2015, "Manning formula", https://en.wikipedia.org/wiki/Manning_formula.

6. John C. Crittenden, R. Rhodes Trussell, David W. Hand, Kerry J. Howe, George Tchobanoglous, *MWH's Water Treatment: Principles and Design*, Third Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2012, page 1861-1862.

7. Andrew Chadwick, John Morfett and Martin Borthwick, *Hydraulics in Civil and Environmental Engineering*, Fourth Edition, New York City, New York: Spon Press, Inc., 2004, page 133.

8. Robert L. Mott and Joseph A. Untener, *Applied Fluid Mechanics*, Seventh Edition, New York City, New York: Pearson, 2015, page 376, 379-380.

9. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 21, 40-41.

10. Gary P. Merkley, "BIE6300 - Irrigation & Conveyance Control Systems, Spring 2004", 2004, Biological and Irrigation Engineering - OCW. Course 2, https://digitalcommons.usu.edu/ocw_bie/2/.

11. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://physics.nist.gov/cgi-bin/cuu/Value?gn.

12. Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, "Conversion of units", https://en.wikipedia.org/wiki/Conversion_of_units.

See Also

`Manningtrap` for a trapezoidal cross-section, `Manningtri` for a triangular cross-section, `Manningpara` for a parabolic cross-section, and `Manningcirc` for a circular cross-section.

`Manningtrap` *Trapezoidal cross-section for the Gauckler-Manning-Strickler equation*

Description

This function solves for one missing variable in the Gauckler-Manning- Strickler equation for a trapezoidal cross-section and uniform flow. The `uniroot` function is used to obtain the missing parameters.

Usage

```
Manningtrap(
  Q = NULL,
  n = NULL,
  m = NULL,
  m1 = NULL,
  m2 = NULL,
  Sf = NULL,
  y = NULL,
  b = NULL,
  Temp = NULL,
  units = c("SI", "Eng"),
  type = c("symmetrical", "non-symmetrical"),
  output = c("list", "data.table")
)
```

Arguments

Q	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
n	numeric vector that contains the Manning's roughness coefficient n, if known.
m	numeric vector that contains the symmetric "cross-sectional side slope of m:V (horizontal:vertical)", if known.
m1	numeric vector that contains the non-symmetric "cross-sectional side slope of m1:V (horizontal:vertical)", if known.
m2	numeric vector that contains the non-symmetric "cross-sectional side slope of m2:V (horizontal:vertical)", if known.
Sf	numeric vector that contains the bed slope (m/m or ft/ft), if known.
y	numeric vector that contains the flow depth (m or ft), if known.
b	numeric vector that contains the bottom width, if known.
Temp	numeric vector that contains the temperature (degrees C or degrees Fahrenheit), if known. Otherwise, the default value is 20 degrees Celsius (68 degrees Fahrenheit).
units	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]
type	character vector that contains the type of trapezoid (symmetrical or non-symmetrical). The symmetrical trapezoid uses `m` while the non- symmetrical trapezoid uses `m1` and `m2`.
output	character vector that contains the output type, either it will be a `list` or `data.table`. The list is the easiest to obtain a singular value. Please see the examples and the vignettes.

Details

Parameters Definitions from Chow pages 7, 13, 20, 22-23 "The depth of flow y is the vertical distance of the lowest point of a channel section from the free surface."

"The top width Temp is the width of channel section at the free surface."

"The water area A is water area of the flow normal to the direction of flow."

"The wetted perimeters P is the length of the line of intersection of the channel wetted surface with a cross-sectional plane normal to the direction of flow."

"The hydraulic radius R is the ratio of the water area to its wetted perimeters."

"The hydraulic radius D is the ratio of the water area to the top width."

"The section factor for critical-flow computation Z is the product of the water area and the square root of the hydraulic depth."

"The section factor for uniform-flow computation AR^2/3 is the product of the water area and the two-thirds power of the hydraulic radius."

"A channel built with unvarying cross section and constant bottom slope is called a prismatic channel. Otherwise, the channel is nonprismatic."

"For any flow, the discharge Q at a channel section is expressed by Q = V A where V is the mean velocity and A is the flow cross-sectional area normal to the direction of the flow, since the mean velocity is defined as the discharge divided by the cross-sectional area."

"The effect of viscosity relative to inertia can be represented by the Reynolds number. ..."

"The effect of gravity upon the state of flow is represented by a ratio of inertial forces to gravity forces. This ratio is given by the Froude number. ..."

The References for the following equations, include, but are not limited to: Chow pages 5, 7, 13, 21, 23, 28; Schall pages 4-17 and 5-5; Wikimedia Conversion and Manning

Gauckler-Manning-Strickler equation is expressed as

$$V = \frac{K_n}{n} R^{\frac{2}{3}} S^{\frac{1}{2}}$$

V the mean velocity (m/s or ft/sec)

n Manning's roughness coefficient (dimensionless)

R the hydraulic radius (m or ft)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

This equation is also expressed as

$$Q = \frac{K_n}{n} \frac{A^{\frac{5}{3}}}{P^{\frac{2}{3}}} S^{\frac{1}{2}}$$

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

n Manning's roughness coefficient (dimensionless)

P the wetted perimeters of the channel (m or ft)

A water area (m^2 or ft^2)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

Other important equations regarding the trapezoidal cross-section follow:

$$R = \frac{A}{P}$$

R the hydraulic radius (m or ft)

A water area (m^2 or ft^2)

P the wetted perimeters of the channel (m or ft)

$$A = y\,(b + my)$$

A water area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

b the bottom width (m or ft)

$$P = b + 2y\sqrt{(1 + m^2)}$$

P the wetted perimeters of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

b the bottom width (m or ft)

$$B = b + 2my$$

B the top width of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

b the bottom width (m or ft)

$$D = \frac{A}{B}$$

D the hydraulic depth (m or ft)

A water area (m^2 or ft^2)

B the top width of the channel (m or ft)

$$Z = \frac{[(b + my)y]^1.5}{\sqrt{b + 2my}}$$

Z the Section factor (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

b the bottom width (m or ft)

$$E = y + \frac{Q^2}{2gA^2}$$

E the Specific Energy (m or ft)

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

g gravitational acceleration (m/s^2 or ft/sec^2)

A water area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

$$VH = \frac{V^2}{2g}$$

VH the Velocity Head (m or ft)

V the mean velocity (m/s or ft/sec)

g gravitational acceleration (m/s^2 or ft/sec^2)

$$w = y\sqrt{m^2 + 1}$$

w the Wetted Length (m or ft)

m the horizontal side slope

y the flow depth (normal depth in this function) [m or ft]

$$tau_0 = gammaRS$$

\tau_0 "mean boundary shear stress" (N/m^2 or lbf/ft^2)

\gamma unit weight of water at the given temperature (N/m^3 or lbf/ft^3)

R the hydraulic radius (m or ft)

S the slope of the channel bed ["average bottom slope (equal to energy slope for uniform flow)"] (m/m or ft/ft)

$$tau_d = gammayS$$

\tau_d "shear stress in channel at maximum depth" (N/m^2 or lbf/ft^2)

\gamma unit weight of water at the given temperature (N/m^3 or lbf/ft^3)

y the flow depth ("maximum depth of flow in the channel for the design discharge") [m or ft]

S the slope of the channel bed ["average bottom slope (equal to energy slope for uniform flow)"] (m/m or ft/ft)

where

$$gamma = grho$$

\gamma unit weight of water at the given temperature (N/m^3 or lbf/ft^3)

g gravitational acceleration (m/s^2 or ft/sec^2)

\rho density of the fluid at a certain temperature (kg/m^3 or slugs/ft^3)

$$gamma = rho\frac{g}{g_c}$$

\gamma unit weight of water at the given temperature (lbf/ft^3)

\rho density of the fluid at a certain temperature (lbm/ft^3)

g gravitational acceleration (m/s^2 or ft/sec^2)

gc gravitational constant (32.2 lbm-ft/lbf-sec^2) used for dimensional analysis so that the Reynolds number will be dimensionless with US Customary units

$$K = \frac{k(A * R^(2/3))}{n}$$

K channel conveyance (m^3/s or ft^3/sec)

k unit conversion factor (1 in SI and 3.2808399^(1/3) in US Customary units

A water area (m^2 or ft^2)

R the hydraulic radius (m or ft)

n Manning's roughness coefficient (dimensionless)

A rough turbulent zone check is performed on the water flowing in the channel using the Reynolds number (Re). The Re equation follows:

$$Re = \frac{rhoRV}{mu}$$

Re Reynolds number (dimensionless)

\rho water density (kg/m^3 or slug/ft^3)

R the hydraulic radius (m or ft)

V the mean velocity (m/s or ft/sec)

\mu dynamic viscosity (* 10^-3 kg/m*s or * 10^-5 lb*sec/ft^2)

A critical flow check is performed on the water flowing in the channel using the Froude number (Fr). The Fr equation follows:

$$Fr = \frac{V}{(\sqrt{g * D})}$$

Fr the Froude number (dimensionless)

V the mean velocity (m/s or ft/sec)

g gravitational acceleration (m/s^2 or ft/sec^2)

D the hydraulic depth (m or ft)

Value

the missing parameters (Q, n, b, m, m1, m2, Sf, or y) & V (velocity), Flow depth (y), Bottom width (b), symmetric side slope (m), Slope (Sf), A (area), P (wetted perimeters), R (hydraulic radius), B (top width), D (hydraulic depth), w (Wetted Length), w1 (Wetted Length for a non-symmetric trapezoid), w2 (Wetted Length for a non-symmetric trapezoid), Z (Section Factor), E (Specific Energy), K (conveyance), Vel_Head (Velocity Head), Re (Reynolds number), Fr (Froude number), taud (maximum shear stress), tau0 (average shear stress) as a `list`. Alternatively, the Flow depth (y), Flow area (A), Wetted Perimeters (P), Top Width (B), Bottom width (b), Hydraulic Radius (R), Hydraulic Depth (D), Flow Mean Velocity (V), Flow Discharge (Q), Manning's roughness coefficient (n), Slope (Sf), Temperature, Absolute Temperature, Saturated Liquid Density, Absolute or Dynamic Viscosity, Kinematic Viscosity, Froude number (Fr), Reynolds number (Re), symmetric side slope (m), non-symmetric side slope (m1), non-symmetric side slope (m2), Wetted Length (w), Wetted Length for a non-symmetric trapezoid (w1), Wetted Length for a non-symmetric trapezoid (w2), Section Factor (Z), conveyance (K), Specific Energy (E), Velocity Head (Vel_Head), Maximum Shear Stress (taud), Average Shear Stress (tau0) along with the associated units can be returned in a `data.table`.

Note

Assumption: Surface water temperature of 20 degrees Celsius (68 degrees Fahrenheit) at atmospheric pressure

Note: Units must be consistent

Author(s)

Irucka Embry

References

1. Harlan Bengtson, "Calculation of Open Channel Flow Hydraulic Radius: Calculate using Trapezoid Area", Bright Hub Engineering Hydraulics in Civil Engineering, https://www.brighthubengineering.com/hydraulics-civil-engineering/67126-calculation-of-hydraulic-radius-fo

2. Andrew Chadwick, John Morfett, and Martin Borthwick, *Hydraulics in Civil and Environmental Engineering*, Fourth Edition, New York City, New York: Spon Press, 2004, pages 132-133.

3. R.J. Charbeneau, "Topic 8: Open Channel Flow", CE 365K Hydraulic Engineering Design, The University of Texas at Austin Cockrell School of Engineering Department of Civil, Architectural and Environmental Engineering, https://www.caee.utexas.edu/prof/maidment/CE365KSpr14/Visual/OpenChannels.pdf.

4. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 7-8, 13, 20-23, 28, 39-43.

5. John C. Crittenden, R. Rhodes Trussell, David W. Hand, Kerry J. Howe, George Tchobanoglous, *MWH's Water Treatment: Principles and Design*, Third Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2012, pages 1861-1862.
6. Prof. Dr. Aminuddin Ab Ghani, "Specific Energy & Hydraulic Jump", Universiti Sains Malaysia (USM) Engineering Campus River Engineering and Urban Drainage Research Centre (REDACE), https://web.archive.org/web/20200110165556/https://redac.eng.usm.my/EAH/Handouts/Specific%20Energy%202011.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
7. Tyler G. Hicks, P.E., *Civil Engineering Formulas: Pocket Guide*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2002, pages 423, 425.
8. Gary P. Merkley, "BIE6300 - Irrigation & Conveyance Control Systems, Spring 2004", 2004, Biological and Irrigation Engineering - OCW. Course 2, https://digitalcommons.usu.edu/ocw_bie/2/.
9. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
10. Robert L. Mott and Joseph A. Untener, *Applied Fluid Mechanics*, Seventh Edition, New York City, New York: Pearson, 2015, pages 376, 392.
11. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://physics.nist.gov/cgi-bin/cuu/Value?gn.
12. James D. Schall, Everett V. Richardson, and Johnny L. Morris, U.S. Department of Transportation Federal Highway Administration & National Highway Institute (NHI) and Office of Bridge Technology (HIBTemp), *Introduction to Highway Hydraulics: Hydraulic Design Series Number 4*, Fourth Edition, June 2008, pages 4-5, 4-16 - 4-17 and 5-5, https://www.fhwa.dot.gov/engineering/hydraulics/pubs/08090/HDS4_608.pdf.
13. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, pages 2, 8, 36, 102, 120, 153.
14. US Department of Transportation Federal Highway Administration (FHWA), "Urban Drainage Design Manual", Hydraulic Engineering Circular No. 22, Third Edition, Publication No. FHWA-NHI-10-009 September 2009 (Revised August 2013), pages 5-7 - 5-8, https://www.fhwa.dot.gov/engineering/hydraulics/library_arc.cfm?pub_number=22&id=140.
15. Ali R.Vatankhah, "Explicit solutions for critical and normal depths in trapezoidal and parabolic open channels", *Ain Shams Engineering Journal*, Volume 4, Issue 1, March 2013, Pages 17-23, https://www.sciencedirect.com/science/article/pii/S2090447912000329.
16. Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, "Conversion of units", https://en.wikipedia.org/wiki/Conversion_of_units.
17. Wikimedia Foundation, Inc. Wikipedia, 23 May 2019, "Manning formula", https://en.wikipedia.org/wiki/Manning_formula.

See Also

Manningrect for a rectangular cross-section, Manningtri for a triangular cross-section, Manningpara for a parabolic cross-section, and Manningcirc for a circular cross-section.

Examples

```
# Example 1

library(iemisc)
# Exercise 4.1 from Sturm (page 153)

uu <- Manningtrap(Q = 3000, b = 40, m = 3, Sf = 0.002, n = 0.025,
units = "Eng", type = "symmetrical", output = "list")
# Q = 3000 cfs, b = 40 ft, m = 3, Sf = 0.002 ft/ft, n = 0.025,
# units = English units
# This will solve for y since it is missing and y will be in ft

uu$y # only returns y

uu # returns all results



# Example 2

# Please refer to the iemisc: Manning... Examples using iemiscdata
# [https://www.ecoccs.com/R_Examples/Manning_iemiscdata_Examples.pdf] and iemisc:
# Open Channel Flow Examples involving Geometric Shapes with the
# Gauckler-Manning-Strickler Equation
# [https://www.ecoccs.com/R_Examples/Open-Channel-Flow_Examples_Geometric_Shapes.pdf]
# for the cross-section examples using iemiscdata
```

Manningtrap_critical — *Trapezoidal cross-section for the Gauckler-Manning-Strickler equation (critical parameters)*

Description

This function solves for one missing variable in the Gauckler-Manning- Strickler equation for a trapezoidal cross-section and uniform flow. The uniroot function is used to obtain the missing parameters. This function provides both normal and critical parameters values.

Usage

```
Manningtrap_critical(
  Q = NULL,
```

```
  n = NULL,
  m = NULL,
  m1 = NULL,
  m2 = NULL,
  Sf = NULL,
  y = NULL,
  b = NULL,
  Temp = NULL,
  units = c("SI", "Eng"),
  type = c("symmetrical", "non-symmetrical"),
  critical = c("approximate", "accurate"),
  output = c("list", "data.table")
)
```

Arguments

Q	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
n	numeric vector that contains the Manning's roughness coefficient n, if known.
m	numeric vector that contains the symmetric "cross-sectional side slope of m:V (horizontal:vertical)", if known.
m1	numeric vector that contains the non-symmetric "cross-sectional side slope of m1:V (horizontal:vertical)", if known.
m2	numeric vector that contains the non-symmetric "cross-sectional side slope of m2:V (horizontal:vertical)", if known.
Sf	numeric vector that contains the bed slope (m/m or ft/ft), if known.
y	numeric vector that contains the flow depth (m or ft), if known.
b	numeric vector that contains the bottom width, if known.
Temp	numeric vector that contains the temperature (degrees C or degrees Fahrenheit), if known.
units	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]
type	character vector that contains the type of trapezoid (symmetrical or non-symmetrical). The symmetrical trapezoid uses `m` while the non- symmetrical trapezoid uses `m1` and `m2`.
critical	character vector that contains the type of critical parameters calculations (`approximate` or `accurate`). The accurate calculation provides parameters where the Froude number is 1. The approximate calculation calculates the values without having the Froude number return 1.
output	character vector that contains the output type, either it will be a `list` or `data.table`. The list is the easiest to obtain a singular value.

Details

Critical State Discussion from Chow pages 13, 63 "When F (Froude number) is equal to unity, ... the flow is said to be in a *critical* state. If F is less than unity, ... the flow is *subcritical*. If F is greater than unity, ... the flow is *supercritical*."

"... the critical state of flow through a channel section is characterized by several important conditions. Recapitulating, they are (1) the specific energy is a minimum for a given discharge; (2) the discharge is a maximum for a given specific energy; (3) the specific force is a minimum for a given discharge; (4) the velocity head is equal to half the hydraulic depth in a channel of small slope; (5) the Froude number is equal to unity; and (6) the velocity of flow in a channel of small slope with uniform velocity distribution is equal to the celerity of small gravity waves in shallow water caused by local disturbances."

"Discussions on critical state of flow have referred mainly to a particular section of a channel, known as the *critical section*. If the critical state of exists throughout the entire length of the channel or over a reach of the channel, the flow in the channel is a *critical flow*."

Value

the missing parameters (Q, n, b, m, m1, m2, Sf, or y) & V (velocity), Flow depth (y), Bottom width (b), symmetric side slope (m), Slope (Sf), A (area), P (wetted perimeters), R (hydraulic radius), B (top width), D (hydraulic depth), w (Wetted Length), w1 (Wetted Length for a non-symmetric trapezoid), w2 (Wetted Length for a non-symmetric trapezoid), Z (Section Factor), E (Specific Energy), K (conveyance), Vel_Head (Velocity Head), Re (Reynolds number), Fr (Froude number), taud (maximum shear stress), tau0 (average shear stress), yc (critical depth), Ac (critical area), Pc (critical wetted perimeters), Bc (critical top width), Rc (critical hydraulic radius), Dc (critical hydraulic depth), Vc (critical velocity), Qc (critical discharge), Sfc (critical slope), Frc (critical Froude number), Zc (critical Section Factor), Ec (critical Specific Energy) as a `list`. Alternatively, the Flow depth (y), Flow area (A), Wetted Perimeters (P), Top Width (B), Bottom width (b), Hydraulic Radius (R), Hydraulic Depth (D), Flow Mean Velocity (V), Flow Discharge (Q), Manning's roughness coefficient (n), Slope (Sf), Temperature, Absolute Temperature, Saturated Liquid Density, Absolute or Dynamic Viscosity, Kinematic Viscosity, Froude number (Fr), Reynolds number (Re), symmetric side slope (m), non-symmetric side slope (m1), non-symmetric side slope (m2), Wetted Length (w), Wetted Length for a non-symmetric trapezoid (w1), Wetted Length for a non-symmetric trapezoid (w2), Section Factor (Z), conveyance (K), Specific Energy (E), Velocity Head (Vel_Head), Maximum Shear Stress (taud), Average Shear Stress (tau0) along with the associated units can be returned in a `data.table`. Both the normal and the critical values (where present) are returned in the table.

Author(s)

Irucka Embry

References

Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 13, 63.

Examples

```
# Example 1

library(iemisc)
# Exercise 4.1 from Sturm (page 153)

uuc <- Manningtrap_critical(Q = 3000, b = 40, m = 3, Sf = 0.002, n = 0.025,
```

```
units = "Eng", type = "symmetrical", critical = "accurate", output = "list")
# Q = 3000 cfs, b = 40 ft, m = 3, Sf = 0.002 ft/ft, n = 0.025,
# units = English units
# This will solve for y since it is missing and y will be in ft

uuc$y # only returns y

uuc # returns all results



# Example 2

# Please refer to the iemisc: Manning... Examples using iemiscdata
# [https://www.ecoccs.com/R_Examples/Manning_iemiscdata_Examples.pdf] and iemisc:
# Open Channel Flow Examples involving Geometric Shapes with the
# Gauckler-Manning-Strickler Equation
# [https://www.ecoccs.com/R_Examples/Open-Channel-Flow_Examples_Geometric_Shapes.pdf]
# for the cross-section examples using iemiscdata
```

Manningtri *Triangular cross-section for the Gauckler-Manning-Strickler equation*

Description

This function solves for one missing variable in the Gauckler-Manning- Strickler equation for a triangular cross-section and uniform flow. The uniroot function is used to obtain the missing parameters.

Usage

```
Manningtri(
  Q = NULL,
  n = NULL,
  m = NULL,
  Sf = NULL,
  y = NULL,
  Temp = NULL,
  units = c("SI", "Eng")
)
```

Arguments

`Q`	numeric vector that contains the discharge value (m^3/s or ft^3/s), if known.
`n`	numeric vector that contains the Manning's roughness coefficient n, if known.
`m`	numeric vector that contains the "cross-sectional side slope of m:1 (horizontal:vertical)", if known.
`Sf`	numeric vector that contains the bed slope (m/m or ft/ft), if known.
`y`	numeric vector that contains the flow depth (m or ft), if known.
`Temp`	numeric vector that contains the temperature (degrees C or degrees Fahrenheit), if known.
`units`	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]

Details

Gauckler-Manning-Strickler equation is expressed as

$$V = \frac{K_n}{n} R^{\frac{2}{3}} S^{\frac{1}{2}}$$

V the velocity (m/s or ft/s)

n Manning's roughness coefficient (dimensionless)

R the hydraulic radius (m or ft)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

This equation is also expressed as

$$Q = \frac{K_n}{n} \frac{A^{\frac{5}{3}}}{P^{\frac{2}{3}}} S^{\frac{1}{2}}$$

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

n Manning's roughness coefficient (dimensionless)

P the wetted perimeters of the channel (m or ft)

A the cross-sectional area (m^2 or ft^2)

S the slope of the channel bed (m/m or ft/ft)

K_n the conversion constant – 1.0 for SI and 3.2808399 ^ (1 / 3) for English units – m^(1/3)/s or ft^(1/3)/s

Other important equations regarding the triangular cross-section follow:

$$R = \frac{A}{P}$$

R the hydraulic radius (m or ft)

A the cross-sectional area (m^2 or ft^2)

P the wetted perimeters of the channel (m or ft)

$$A = my^2$$

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$P = 2y\sqrt{(1 + m^2)}$$

P the wetted perimeters of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$B = 2my$$

B the top width of the channel (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$D = \frac{A}{B}$$

D the hydraulic depth (m or ft)

A the cross-sectional area (m^2 or ft^2)

B the top width of the channel (m or ft)

$$Z = \frac{\sqrt{2}}{2}my^2.5$$

Z the Section factor (m or ft)

y the flow depth (normal depth in this function) [m or ft]

m the horizontal side slope

$$E = y + \frac{Q^2}{2gA^2}$$

E the Specific Energy (m or ft)

Q the discharge [m^3/s or ft^3/s (cfs)] is VA

g gravitational acceleration (m/s^2 or ft/sec^2)

A the cross-sectional area (m^2 or ft^2)

y the flow depth (normal depth in this function) [m or ft]

$$VH = \frac{V^2}{2g}$$

VH the Velocity Head (m or ft)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

$$w = \sqrt{y^2 + (y * m)^2}$$

w the Wetted Length (m or ft)

m the horizontal side slope

y the flow depth (normal depth in this function) [m or ft]

A rough turbulent zone check is performed on the water flowing in the channel using the Reynolds number (Re). The Re equation follows:

$$Re = \frac{rhoRV}{mu}$$

Re Reynolds number (dimensionless)

\rho density (kg/m^3 or slug/ft^3)

R the hydraulic radius (m or ft)

V the velocity (m/s or ft/s)

\mu dynamic viscosity (* 10^-3 kg/m*s or * 10^-5 lb*s/ft^2)

A critical flow check is performed on the water flowing in the channel using the Froude number (Fr). The Fr equation follows:

$$Fr = \frac{V}{\left(\sqrt{g * D}\right)}$$

Fr the Froude number (dimensionless)

V the velocity (m/s or ft/s)

g gravitational acceleration (m/s^2 or ft/sec^2)

D the hydraulic depth (m or ft)

Value

the missing parameter (Q, n, m, Sf, or y) & area (A), wetted perimeter (P), velocity (V), top width (B), hydraulic radius (R), Reynolds number (Re), and Froude number (Fr) as a `list`.

Note

Assumptions: uniform flow, prismatic channel, and surface water temperature of 20 degrees Celsius (68 degrees Fahrenheit) at atmospheric pressure

Note: Units must be consistent

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 2, 8, 36, 102, 120, 153-154.
2. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
3. Gilberto E. Urroz, Utah State University Civil and Environmental Engineering - OCW, CEE6510 - Numerical Methods in Civil Engineering, Spring 2006 (2006). Course 3. "Solving selected equations and systems of equations in hydraulics using Matlab", August/September 2004, https://digitalcommons.usu.edu/ocw_cee/3/.
4. Tyler G. Hicks, P.E., *Civil Engineering Formulas: Pocket Guide*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2002, page 423, 425.
5. Wikimedia Foundation, Inc. Wikipedia, 26 November 2015, "Manning formula", https://en.wikipedia.org/wiki/Manning_formula.
6. John C. Crittenden, R. Rhodes Trussell, David W. Hand, Kerry J. Howe, George Tchobanoglous, *MWH's Water Treatment: Principles and Design*, Third Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2012, page 1861-1862.
7. Andrew Chadwick, John Morfett and Martin Borthwick, *Hydraulics in Civil and Environmental Engineering*, Fourth Edition, New York City, New York: Spon Press, Inc., 2004, page 133.
8. Robert L. Mott and Joseph A. Untener, *Applied Fluid Mechanics*, Seventh Edition, New York City, New York: Pearson, 2015, page 376, 393.
9. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 21, 40-41.
10. Gary P. Merkley, "BIE6300 - Irrigation & Conveyance Control Systems, Spring 2004", 2004, Biological and Irrigation Engineering - OCW. Course 2, https://digitalcommons.usu.edu/ocw_bie/2/.
11. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://physics.nist.gov/cgi-bin/cuu/Value?gn.
12. Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, "Conversion of units", https://en.wikipedia.org/wiki/Conversion_of_units.

See Also

Manningtrap for a triangleal cross-section, Manningrect for a rectangular cross-section, Manningpara for a parabolic cross-section, and Manningcirc for a circular cross-section.

Examples

```
# Please refer to the iemisc: Manning... Examples using iemiscdata
# [https://www.ecoccs.com/R_Examples/Manning_iemiscdata_Examples.pdf] and iemisc:
# Open Channel Flow Examples involving Geometric Shapes with the
# Gauckler-Manning-Strickler Equation
# [https://www.ecoccs.com/R_Examples/Open-Channel-Flow_Examples_Geometric_Shapes.pdf]
# for the cross-section examples using iemiscdata



library(iemisc)

# Modified Exercise 4.1 from Sturm (page 153)
Manningtri(Q = 3000, m = 3, Sf = 0.002, n = 0.025, units = "Eng")
# Q = 3000 cfs, m = 3, Sf = 0.002 ft/ft, n = 0.025, units = English units
# This will solve for y since it is missing and y will be in ft



# Modified Exercise 4.5 from Sturm (page 154)
Manningtri(Q = 950, m = 2, Sf = 0.022, n = 0.023, units = "SI")
# Q = 950 m^3/s, m = 2, Sf = 0.022 m/m, n = 0.023, units = SI units
# This will solve for y since it is missing and y will be in m
```

---

maxmre *Maximum Mean relative error (MAXRE)*

Description

This function computes the maximum mean relative error (MAXRE).

Usage

```
maxmre(predicted, observed, na.rm = FALSE)
```

Arguments

predicted	numeric vector that contains the model predicted data points (1st parameters)
observed	numeric vector that contains the observed data points (2nd parameters)
na.rm	logical vector that determines whether the missing values should be removed or not.

Details

(MAXRE) is expressed as

$$max \sum_{i=1}^{N} \left| \frac{P_i - O_i}{O_i} \right|$$

N the number of observations

P_i the predicted values

O_i the observed or reference values

Value

maximum mean relative error (MAXRE) as a numeric vector using the same units as the given variables. The default choice is that any NA values will be kept (`na.rm = FALSE`). This can be changed by specifying `na.rm = TRUE`, such as `maxmre(pre, obs, na.rm = TRUE)`.

Author(s)

Irucka Embry

References

Huang, J. (2018). "A Simple Accurate Formula for Calculating Saturation Vapor Pressure of Water and Ice", *Journal of Applied Meteorology and Climatology*, 57(6), 1265-1272. Retrieved Nov 4, 2021, https://web.archive.org/web/20221024040058/https://journals.ametsoc.org/view/journals/apme/57/6/jamc-d-17-0334.1.xml. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.

See Also

`mape` for mean absolute percent error (MAPE), `madstat` for mean-absolute deviation (MAD), `dr` for "index of agreement (dr)", `vnse` for Nash-Sutcliffe model efficiency (NSE), `rmse` for root mean square error (RMSE), and `mre` for the mean relative error (MRE).

Examples

```
# Example 1

library(iemisc)

obs <- 1:10 # observed
pre <- 2:11 # predicted
maxmre(pre, obs)



# Example 2
```

```
install.load::load_package("iemisc", "rando", "data.table")

set_n(100) # makes the example reproducible
obs1 <- r_norm(.seed = 605) # observed
pre1 <- r_norm(.seed = 364) # predicted


# using the vectors pre1 and obs1
maxmre(pre1, obs1)


# using a matrix of the numeric vectors pre1 and obs1
mat1 <- matrix(data = c(obs1, pre1), nrow = length(pre1), ncol = 2,
byrow = FALSE, dimnames = list(c(rep("", length(pre1))),
c("Predicted", "Observed")))
maxmre(mat1[, 2], mat1[, 1])

# mat1[, 1] # observed values from column 1 of mat1
# mat1[, 2] # predicted values from column 2 of mat1


# using a data.frame of the numeric vectors pre1 and obs1
df1 <- data.frame(obs1, pre1)
maxmre(df1[, 2], df1[, 1])

# df1[, 1] # observed values from column 1 of df1
# df1[, 2] # predicted values from column 2 of df1



# using a data.table of the numeric vectors pre1 and obs1
df2 <- data.table(obs1, pre1)
maxmre(df2[, 2, with = FALSE][[1]], df2[, 1, with = FALSE][[1]])

# df2[, 1, with = FALSE][[1]] # observed values from column 1 of df2
# df2[, 2, with = FALSE][[1]] # predicted values from column 2 of df2
```

Mod_octave *Modulus Operation (GNU Octave/MATLAB compatible)*

Description

Obtain the modulo of x and y in a manner compatible with GNU Octave/MATLAB.

Usage

```
Mod_octave(x, n)
```

Arguments

x, n	An R object (array, matrix, vector)

Value

Return the computed modulo of x and y.

Author(s)

Irucka Embry, Samit Basu (FreeMat)

References

1. Samit Basu (2002-2006). FreeMat v4.0, https://freemat.sourceforge.net/help/mathfunctions_mod.html.
2. John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 564.

Examples

```
# Example from FreeMat

library(iemisc)

Mod_octave(18, 12)

# Please refer to the iemisc: Examples from GNU Octave Rem, Mod, and
# fractdiff Compatible Functions vignette for additional examples
```

---

mortality_rate *Mortality Rate*

Description

This function calculates the mortality rate which is also known as the crude death rate for a given population.

Usage

```
mortality_rate(number_people_dead, population_size, n)
```

Arguments

`number_people_dead`
: numeric vector that contains the number of people dead for a time period

`population_size`
: numeric vector that contains the total population size of which the deaths occurred

`n`
: numeric vector that contains the population size units (ex., 3 for 1,000 people, 5 for 100,000 people)

Value

the mortality rate as a numeric vector

Author(s)

Irucka Embry

References

1. Giovanni Scerra, Published Sep 26, 2021, "The Math of the Pandemic: COVID-19 Mortality Rate", LinkedIn, https://www.linkedin.com/pulse/math-pandemic-covid-19-mortality-rate-giovanni-sce
2. Michael Darcy and Łucja Zaborowska, MD, PhD, Last updated on Nov 05, 2022, "Mortality Rate Calculator", Omni Calculator, https://www.omnicalculator.com/health/mortality-rate.

Examples

```
# Example from Reference 1

library(iemisc)

mortality_rate(369369, 331534662, 5)
```

`mortality_rate_pct` *Mortality Rate Percent*

Description

This function calculates the mortality rate percent which is also known as the crude death rate percent for a given population.

Usage

```
mortality_rate_pct(mortality_rate, n)
```

Arguments

`mortality_rate`	numeric vector that contains the mortality rate
`n`	numeric vector that contains the population size units (ex., 3 for 1,000 people, 5 for 100,000 people)

Value

the mortality rate percent as a numeric vector

Author(s)

Irucka Embry

References

1. Giovanni Scerra, Published Sep 26, 2021, "The Math of the Pandemic: COVID-19 Mortality Rate", LinkedIn, https://www.linkedin.com/pulse/math-pandemic-covid-19-mortality-rate-giovanni-sce
2. Michael Darcy and Łucja Zaborowska, MD, PhD, Last updated on Nov 05, 2022, "Mortality Rate Calculator", Omni Calculator, https://www.omnicalculator.com/health/mortality-rate.

Examples

```
# Example from Reference 1

library(iemisc)

mr_2020 <- mortality_rate(369369, 331534662, 5)

mortality_rate_pct(mortality_rate(15, 331534662, 5), 5)

mortality_rate_pct(mr_2020, 5)

mortality_rate_pct(15, 5)
```

`mre` *Mean relative error (MRE)*

Description

This function computes the mean relative error (MRE).

Usage

```
mre(predicted, observed, na.rm = FALSE)
```

Arguments

`predicted`	numeric vector that contains the model predicted data points (1st parameters)
`observed`	numeric vector that contains the observed data points (2nd parameters)
`na.rm`	logical vector that determines whether the missing values should be removed or not.

Details

(MRE) is expressed as

$$\frac{1}{N}\sum_{i=1}^{N}\left|\frac{P_i - O_i}{O_i}\right|$$

N the number of observations

P_i the predicted values

O_i the observed or reference values

Value

mean relative error (MRE) as a numeric vector using the same units as the given variables. The default choice is that any NA values will be kept (`na.rm = FALSE`). This can be changed by specifying `na.rm = TRUE`, such as `mre(pre, obs, na.rm = TRUE)`.

Author(s)

Irucka Embry

References

Huang, J. (2018). "A Simple Accurate Formula for Calculating Saturation Vapor Pressure of Water and Ice", *Journal of Applied Meteorology and Climatology*, 57(6), 1265-1272. Retrieved Nov 4, 2021, https://web.archive.org/web/20221024040058/https://journals.ametsoc.org/view/journals/apme/57/6/jamc-d-17-0334.1.xml. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.

See Also

mape for mean absolute percent error (MAPE), madstat for mean-absolute deviation (MAD), dr for 'index of agreement (dr)', vnse for Nash-Sutcliffe model efficiency (NSE), rmse for root mean square error (RMSE), and maxmre for the maximum mean relative error (MAXRE).

Examples

```
# Example 1

library(iemisc)

obs <- 1:10 # observed
pre <- 2:11 # predicted
mre(pre, obs)




# Example 2

install.load::load_package("iemisc", "rando")

set_n(100) # makes the example reproducible
obs1 <- r_norm(.seed = 873) # observed
pre1 <- r_norm(.seed = 281) # predicted


# using the vectors pre1 and obs1
mre(pre1, obs1)


# using a matrix of the numeric vectors pre1 and obs1
mat1 <- matrix(data = c(obs1, pre1), nrow = length(pre1), ncol = 2,
   byrow = FALSE, dimnames = list(c(rep("", length(pre1))),
   c("Predicted", "Observed")))
mre(mat1[, 2], mat1[, 1])

# mat1[, 1] # observed values from column 1 of mat1
# mat1[, 2] # predicted values from column 2 of mat1


# using a data.frame of the numeric vectors pre1 and obs1
df1 <- data.frame(obs1, pre1)
mre(df1[, 2], df1[, 1])

# df1[, 1] # observed values from column 1 of df1
# df1[, 2] # predicted values from column 2 of df1


library("data.table")

# using a data.table of the numeric vectors pre1 and obs1
```

```
df2 <- data.table(obs1, pre1)
mre(df2[, 2, with = FALSE][[1]], df2[, 1, with = FALSE][[1]])

# df2[, 1, with = FALSE][[1]] # observed values from column 1 of df2
# df2[, 2, with = FALSE][[1]] # predicted values from column 2 of df2
```

---

n *Manning's n for natural channels*

Description

This function computes Manning's n for natural channels.

Usage

```
n(nb = NULL, n1 = NULL, n2 = NULL, n3 = NULL, n4 = NULL, m = NULL)
```

Arguments

nb	numeric vector that contains "the base value for a straight, uniform channel", if needed
n1	numeric vector that contains "correction for surface irregularities", if needed
n2	numeric vector that contains "correction for variations in the shape and size of the cross section", if needed
n3	numeric vector that contains "correction for obstructions", if needed
n4	numeric vector that contains "correction for vegetation and flow conditions", if needed
m	numeric vector that contains "correction factor for channel meandering", if needed

Details

"Roughness values for channels and flood plains should be determined separately. The composition, physical shape, and vegetation of a flood plain can be quite different from those of a channel." Source: USGS.

The equation to find Manning's n for natural channels is expressed as

$$n = (n_b + n_1 + n_2 + n_3 + n_4)\, m$$

n Manning's n

n_b "the base value for a straight, uniform channel"

n_1 "correction for surface irregularities"

n_2 "correction for variations in the shape and size of the cross section"

n_3 "correction for obstructions"

n_4 "correction for vegetation and flow conditions"

m "correction factor for channel meandering"

Source: Sturm page 114.

Value

n as Manning's n for a natural channel as a numeric vector.

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 114.
2. Guide for Selecting Manning's Roughness Coefficients for Natural Channels and Flood Plains, United States Geological Survey Water-supply Paper 2339 Metric Version
3. George J. Arcement, Jr., and Verne R. Schneider, United States Geological Survey Water-Supply Paper 2339, "Guide for Selecting Manning's Roughness Coefficients for Natural Channels and Flood Plains", 1989, https://pubs.usgs.gov/wsp/2339/report.pdf.

See Also

nc1 for Horton method for composite Manning's n, nc2 for Einstein and Banks method for composite Manning's n, nc3 for Lotter method for composite Manning's n, and nc4 for Krishnamurthy and Christensen method for composite Manning's n.

Examples

```
library(iemisc)

# Example from Table 4. from the USGS Reference text page 35
n(nb = 0.025, n4 = 0.005, m = 1.00)
```

na.interp1 *na.interp1*

Description

This function combines pracma's `interp1` constant interpolation method with zoo's `na.approx` linear interpolation method. Here, `x = x` rather than `x = index(object)` in na.approx. Here, `y = y` rather than `y = object` in na.approx. Also, here, `xi` is used instead of `xout` in na.approx. The Arguments list was obtained from both interp1 and na.approx.

Usage

```
na.interp1(x, y, xi = x, ..., na.rm = TRUE, maxgap = Inf)
```

Arguments

`x`	numeric vector; points on the x-axis; at least two points required; will be sorted if necessary.
`y`	numeric vector; values of the assumed underlying function; `x` and `y` must be of the same length.
`xi`	numeric vector; points at which to compute the interpolation; all points must lie between `min(x)` and `max(x)`.
`...`	further arguments passed to methods. The `n` argument of `approx` is currently not supported.
`na.rm`	logical. If the result of the (`spline`) interpolation still results in `NA`s, should these be removed?
`maxgap`	maximum number of consecutive `NA`s to fill. Any longer gaps will be left unchanged. Note that all methods listed above can accept `maxgap` as it is ultimately passed to the default method.

Value

Numeric vector representing values at points `xi`.

Author(s)

Hans Werner Borchers (pracma interp1), Felix Andrews (zoo na.approx), Irucka Embry

Source

1. zoo's na.approx.R - modified on Fri Aug 6 00:26:22 2010 UTC by felix. See https://r-forge.r-project.org/scm/viewvc.php/pkg/zoo/R/na.approx.R?view=markup&revision=781&root=zoo.
2. pracma interp1 function definition - R package pracma created and maintained by Hans Werner Borchers. See `interp1`.

See Also

na.approx, interp1

Examples

```
# zoo time series example

install.load::load_package("iemisc", "data.table")

zoo1 <- structure(c(1.6, 1.7, 1.7, 1.7, 1.7, 1.7, 1.6, 1.7, 1.7, 1.7,
1.7, 1.7, 2, 2.1, 2.1, NA, NA, 2.1, 2.1, NA, 2.3, NA, 2, 2.1), .Dim = c(12L,
2L), .Dimnames = list(NULL, c("V1", "V2")), index = structure(c(1395242100,
1395243000, 1395243900, 1395244800, 1395245700, 1395256500, 1395257400,
1395258300, 1395259200, 1395260100, 1395261000, 1395261900), class =
c("POSIXct", "POSIXt"), tzone = "GMT"), class = "zoo")

zoo1 <- as.data.frame(zoo1) # to data.frame from zoo

zoo1[, "Time"] <- as.POSIXct(rownames(zoo1)) # create column named Time as a
# POSIXct class

zoo1 <- setDT(zoo1) # create data.table out of data.frame

setcolorder(zoo1, c(3, 1, 2)) # set the column order as the 3rd column
# followed by the 2nd and 1st columns

zoo1 <- setDF(zoo1) # return to data.frame

rowsinterps1 <- which(is.na(zoo1$V2 == TRUE))

# index of rows of zoo1 that have NA (to be interpolated)
xi <- as.numeric(zoo1[which(is.na(zoo1$V2 == TRUE)), 1])

# the Date-Times for V2 to be interpolated in numeric format
interps1 <- na.interp1(as.numeric(zoo1$Time), zoo1$V2, xi = xi,
na.rm = FALSE, maxgap = 1)

# the interpolated values where only gap sizes of 1 are filled
zoo1[rowsinterps1, 3] <- interps1

# replace the NAs in V2 with the interpolated V2 values
zoo1





# data frame time series example

library(iemisc)
```

```
df1 <- structure(list(Time = structure(c(1395242100, 1395243000, 1395243900,
 1395244800, 1395245700, 1395256500, 1395257400, 1395258300, 1395259200,
 1395260100, 1395261000, 1395261900), class = c("POSIXct", "POSIXt"),
 tzone = "GMT"), V1 = c(1.6, 1.7, 1.7, 1.7, 1.7, 1.7, 1.6, 1.7, 1.7, 1.7,
 1.7, 1.7), V2 = c(2, 2.1, 2.1, NA, NA, 2.1, 2.1, NA, 2.3, NA, 2, 2.1)),
 .Names = c("Time", "V1", "V2"), row.names = c(NA, -12L),
 class = "data.frame")

rowsinterps1 <- which(is.na(df1$V2 == TRUE))

# index of rows of df1 that have NA (to be interpolated)
xi <- as.numeric(df1[which(is.na(df1$V2 == TRUE)), 1])

# the Date-Times for V2 to be interpolated in numeric format
interps1 <- na.interp1(as.numeric(df1$Time), df1$V2, xi = xi,
 na.rm = FALSE, maxgap = 1)

# the interpolated values where only gap sizes of 1 are filled
df1[rowsinterps1, 3] <- interps1

# replace the NAs in V2 with the interpolated V2 values
df1





# data.table time series example

install.load::load_package("iemisc", "data.table")

dt1 <- structure(list(Time = structure(c(1395242100, 1395243000, 1395243900,
 1395244800, 1395245700, 1395256500, 1395257400, 1395258300, 1395259200,
 1395260100, 1395261000, 1395261900), class = c("POSIXct", "POSIXt"),
 tzone = "GMT"), V1 = c(1.6, 1.7, 1.7, 1.7, 1.7, 1.7, 1.6, 1.7, 1.7, 1.7,
 1.7, 1.7), V2 = c(2, 2.1, 2.1, NA, NA, 2.1, 2.1, NA, 2.3, NA, 2, 2.1)),
 .Names = c("Time", "V1", "V2"), row.names = c(NA, -12L), class =
 c("data.table", "data.frame"), sorted = "Time")

rowsinterps2 <- which(is.na(dt1[, 3, with = FALSE] == TRUE))

# index of rows of x that have NA (to be interpolated)
xi <- as.numeric(dt1[rowsinterps2, Time])

# the Date-Times for V2 to be interpolated in numeric format
interps2 <- dt1[, na.interp1(as.numeric(Time), V2, xi = xi,
 na.rm = FALSE, maxgap = 1)]

# the interpolated values where only gap sizes of 1 are filled
dt1[rowsinterps2, `:=` (V2 = interps2)]

# replace the NAs in V2 with the interpolated V2 values
```

```
dt1
```

---

nc1 *Horton method for composite Manning's n*

Description

This function computes the composite Manning's n using the Horton method.

Usage

```
nc1(P, n)
```

Arguments

P	numeric vector that contains "wetted perimeters of any section i"
n	numeric vector that contains "Manning's n of any section i"

Details

"A composite value of Manning's n for a single channel; that is, for the main channel only of a compound channel or a canal with laterally varying roughness." Source: Sturm page 118.

The equation to find Manning's composite n using the Horton method is

$$n_c = \left[\frac{\sum\limits_{i=1}^{N} P_i n_i^{\frac{3}{2}}}{P} \right]^{\frac{2}{3}}$$

n_c Manning's composite n

P "wetted perimeters of the entire cross section"

P_i "wetted perimeters of any section i"

n_i "Manning's n of any section i"

N "total number of sections into which the wetted perimeters is divided"

Source: Sturm page 118.

Value

numeric vector that contains nc1 as Manning's composite n.

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 118.
2. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

See Also

n for Manning's n for natural channels, nc2 for Einstein and Banks method for composite Manning's n, nc3 for Lotter method for composite Manning's n, and nc4 for Krishnamurthy and Christensen method for composite Manning's n.

Examples

```
library(iemisc)

# Example from the Moore Reference text
nc1(n = c(0.05, 0.035, 0.05, 0.04), P = c(22.22, 34.78, 2.00, 6.08))
```

---

nc2 *Einstein and Banks method for composite Manning's n*

Description

This function computes the composite Manning's n using the Einstein and Banks method.

Usage

```
nc2(P, n)
```

Arguments

P	numeric vector that contains "wetted perimeters of any section i"
n	numeric vector that contains "Manning's n of any section i"

Details

"A composite value of Manning's n for a single channel; that is, for the main channel only of a compound channel or a canal with laterally varying roughness." Source: Sturm page 118.

The equation to find Manning's composite n using the Einstein and Banks method is

$$n_c = \left[\frac{\sum\limits_{i=1}^{N} P_i n_i^2}{P} \right]^{\frac{1}{2}}$$

n_c Manning's composite n

P "wetted perimeters of the entire cross section"

P_i "wetted perimeters of any section i"

n_i "Manning's n of any section i"

N "total number of sections into which the wetted perimeters is divided"

Source: Sturm page 118.

Value

numeric vector that contains nc2 as Manning's composite n.

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 118-119.
2. Dan Moore, P.E., NRCS Water Quality and Quantity Technology Development Team, Portland Oregon, "Using Mannings Equation with Natural Streams", August 2011, https://web.archive.org/web/20210416091858/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/xsec/manningsNaturally.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

See Also

n for Manning's n for natural channels, nc1 for Horton method for composite Manning's n, nc3 for Lotter method for composite Manning's n, and nc4 for Krishnamurthy and Christensen method for composite Manning's n.

Examples

```
library(iemisc)

# Example from the Moore Reference text
nc2(n = c(0.05, 0.035, 0.05, 0.04), P = c(22.22, 34.78, 2.00, 6.08))
```

---

nc3 *Lotter method for composite Manning's n*

Description

This function computes the composite Manning's n using the Lotter method.

Usage

```
nc3(P, n, R)
```

Arguments

P	numeric vector that contains "wetted perimeters of any section i"
n	numeric vector that contains "Manning's n of any section i"
R	numeric vector that contains "hydraulic radius of any section i"

Details

"A composite value of Manning's n for a single channel; that is, for the main channel only of a compound channel or a canal with laterally varying roughness."

The equation to find Manning's composite n using the Lotter method is

$$n_c = \frac{PR^{\frac{5}{3}}}{\sum\limits_{i=1}^{N} \frac{P_i R_i^{\frac{5}{3}}}{n_i}}$$

n_c Manning's composite n

P "wetted perimeters of the entire cross section"

R "hydraulic radius of the entire cross section"

P_i "wetted perimeters of any section i"

R_i "hydraulic radius of any section i"

n_i "Manning's n of any section i"

N "total number of sections into which the wetted perimeters and hydraulic radius are divided"

Value

numeric vector that contains nc3 as Manning's composite n.

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 118-119.

See Also

`n` for Manning's n for natural channels, `nc1` for Horton method for composite Manning's n, `nc2` for Einstein and Banks method for composite Manning's n, and `nc4` for Krishnamurthy and Christensen method for composite Manning's n.

Examples

```
library(iemisc)

nc3(n = c(0.0024, 0.035), P = c(23.65, 36.08), R = c(2.02, 6.23))
```

---

nc4 — *Krishnamurthy and Christensen method for composite Manning's n*

Description

This function computes the composite Manning's n using the Krishnamurthy and Christensen method.

Usage

```
nc4(P, n, y)
```

Arguments

`P`	numeric vector that contains "wetted perimeters of any section i"
`n`	numeric vector that contains "Manning's n of any section i"
`y`	numeric vector that contains "flow depth in the ith section"

Details

"A composite value of Manning's n for a single channel; that is, for the main channel only of a compound channel or a canal with laterally varying roughness."

The equation to find Manning's composite n using the Krishnamurthy and Christensen method is

$$\ln n_c = \frac{\sum\limits_{i=1}^{N} P_i y_i^{\frac{3}{2}} \ln n_i}{\sum\limits_{i=1}^{N} P_i y_i^{\frac{3}{2}}}$$

n_c Manning's composite n

P_i "wetted perimeters of any section i"

y_i "flow depth in the ith section"

n_i "Manning's n of any section i"

N "total number of sections into which the wetted perimeters and hydraulic radius are divided"

Value

numeric vector that contains nc4 as Manning's composite n.

Author(s)

Irucka Embry

References

1. Terry W. Sturm, *Open Channel Hydraulics*, 2nd Edition, New York City, New York: The McGraw-Hill Companies, Inc., 2010, page 118-119.

See Also

n for Manning's n for natural channels, nc1 for Horton method for composite Manning's n, nc2 for Einstein and Banks method for composite Manning's n, and nc3 for Lotter method for composite Manning's n.

Examples

```
library(iemisc)

nc4(n = c(0.0024, 0.035), P = c(23.65, 36.08), y = c(10.23, 7.38))
```

`ndims` *Number of dimensions in an Array (GNU Octave/MATLAB compatible)*

Description

Obtain the number of dimensions of an array [arrays, matrices, and vectors] in a manner compatible with GNU Octave/MATLAB.

Usage

```
ndims(x)
```

Arguments

x	An array (array, matrix, vector)

Value

"Return the number of dimensions of a. For any array, the result will always be greater than or equal to 2. Trailing singleton dimensions are not counted." Source: Eaton page 46.

Author(s)

Irucka Embry, Samit Basu (FreeMat)

References

1. Samit Basu (2002-2006). FreeMat v4.0, https://freemat.sourceforge.net/help/inspection_ndims.html.
2. John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 46.

See Also

size

Examples

```
library(iemisc)

# Examples from GNU Octave ndims

b <- matlab::ones(c(4, 1, 2, 1))

ndims(b)
```

`ngivenPFi` *To Find i Given F, n, and P (Engineering Economics)*

Description

Compute n given P, F, and i

Usage

```
ngivenPFi(P, F, i)
```

Arguments

`P`	numeric vector that contains the present value(s)
`F`	numeric vector that contains the future value(s)
`i`	numeric vector that contains the interest rate(s) as a percent

Details

n is expressed as

$$n = \frac{\log\left(\frac{F}{P}\right)}{\log\left(1 + i\right)}$$

n the "number of interest periods"

F the "future equivalent"

P the "present equivalent"

i the "effective interest rate per interest period"

Value

n numeric vector that contains the period value(s)

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 129, 142.

Examples

```
# Example for equation 4-7 from the Reference text (page 142)

library(iemisc)

ngivenPFi(P = 500, F = 1000, i = 15)
```

---

numel *Number of elements (GNU Octave/MATLAB compatible)*

Description

Obtain the number of elements of R objects [arrays, matrices, and vectors (including lists)] in a manner compatible with GNU Octave/MATLAB. Some documentation from length.

Usage

```
numel(x, ...)
```

Arguments

x	An R object (array, matrix, vector)
...	R objects (indices idx1, idx2, ...)

Value

"Return the number of elements in the R object x. Optionally, if indices idx1, idx2, ... are supplied, return the number of elements that would result from the indexing a(idx1, idx2, ...)." Source: Eaton page 41.

Author(s)

Irucka Embry, Samit Basu (FreeMat)

Source

1. r - Add a Column to a Dataframe From a List of Values - Stack Overflow answered by Matthew Plourde on Jun 21 2012. See https://stackoverflow.com/questions/11130037/add-a-column-to-a-dataframe 11130178.
2. r - Why does is.vector() return TRUE for list? - Stack Overflow answered by Andrie on May 17 2011. See https://stackoverflow.com/questions/6032772/why-does-is-vector-return-true-for-list/ 6032909.

References

1. Samit Basu (2002-2006). FreeMat v4.0, https://freemat.sourceforge.net/help/inspection_numel.html.
2. John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Pages 46-47.

See Also

numel, numel, size, length, length_octave

Examples

```
library(iemisc)

import::from(matlab, ones)

xx <- list(1:26, 1:10)

numel(xx)
```

PgivenA *Present value given Annual value (Engineering Economics)*

Description

Compute P given A

Usage

```
PgivenA(
  A,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)

PA(
  A,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

`A`	numeric vector that contains the annual value(s)
`n`	numeric vector that contains the period value(s)
`i`	numeric vector that contains the interest rate(s) as a percent
`frequency`	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

P is expressed as

$$P = A \left[\frac{(1+i)^n - 1}{i\,(1+i)^n} \right]$$

P the "present equivalent"

A the "uniform series amount (occurs at the end of each interest period)"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

PgivenA numeric vector that contains the present value(s) rounded to 2 decimal places

PA data.table of both n (0 to n) and the resulting present values rounded to 2 decimal places

Author(s)

Irucka Embry

Source

1. r - Convert column classes in data.table - Stack Overflow answered by Matt Dowle on Dec 27 2013. See https://stackoverflow.com/questions/7813578/convert-column-classes-in-data-table.
2. r - foreach loop not replicating traditional loop - Stack Overflow answered by F. Privé on Oct 19 2019. See https://stackoverflow.com/questions/58459665/r-foreach-loop-not-replicating-traditional-loop.

References

1. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 133-134, 142, 164.
2. Dave Bruns, Exceljet: "Calculate original loan amount", `https://exceljet.net/formulas/calculate-original-loan-amount`.

Examples

```
library(iemisc)

# Example 1 -- Example 4-9 from the Sullivan Reference text (page 133-134)
PgivenA(A = 20000, n = 5, i = 15, frequency = "annual") # the interest rate is 15%

PA(20000, 5, 15, "annual") # the interest rate is 15%


# Example 2

PgivenA(A = 93.22, n = 5, i = 4.50, frequency = "month")

# Using LibreOffice Calc 6.1.5.2 version
# A1 4.50%
# A2 -93.22
# A3 60
# A4 12
# A5 =PV(A1/A4,A3,A2) = $5,000.26
```

PgivenA1 *Present value for geometric gradient series (Engineering Economics)*

Description

Compute P given A1

Usage

```
PgivenA1(A1, i, f, n)
```

Arguments

A1	numeric vector that contains the initial annual value(s)
i	numeric vector that contains the interest rate(s) as a percent
f	numeric vector that contains the average interest rate value(s) as a percent per period
n	numeric vector that contains the period value(s)

Details

P is expressed as

$$P = \frac{A_1 \left[1 - \left(1 + i\right)^{-n} \left(1 + f\right)^n\right]}{i - f}, \; where \; f \neq i$$

or

$$P = A_1 n \left(1 + i\right)^{-1}, \; where \; f = i$$

P "the present equivalent of the geometric gradient series"

A_1 "the initial cash flow in that occurs at the end of period one"

i the "interest rate per period"

f the "average rate each period"

n the "number of interest periods"

Note: "f can be positive or negative"

Value

PgivenA1 numeric vector that contains the present value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 156-159.

Examples

```
library(iemisc)

# Example 4-23 from the Reference text (page 158-159)
PgivenA1(A1 = 1000, i = 25, f = 20, n = 4) # i is 25\% and f is 20\%


# Example 4-24 from the Reference text (page 159)
PgivenA1(A1 = 1000, i = 25, f = -20, n = 4) # i is 25\% and f is -20\%
```

`PgivenAcont` *Present value given Annual value [continuous] (Engineering Economics)*

Description

Compute P given A with interest compounded continuously

Usage

```
PgivenAcont(A, n, r)
```

Arguments

`A`	numeric vector that contains the annual value(s)
`n`	numeric vector that contains the period value(s)
`r`	numeric vector that contains the continuously compounded nominal annual interest rate(s) as a percent

Details

P is expressed as

$$P = A\left[\frac{e^{rn} - 1}{e^{rn}\left(e^{r} - 1\right)}\right]$$

P the "present equivalent"

A the "annual equivalent amount (occurs at the end of each year)"

r the "nominal annual interest rate, compounded continuously"

n the "number of periods (years)"

Value

PgivenAcont numeric vector that contains the present value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 169.

Examples

```
library(iemisc)

PgivenAcont(2000, 3, 12) # the interest rate is 12%
```

PgivenF *Present value given Future value (Engineering Economics)*

Description

Compute P given F

Usage

```
PgivenF(
  F,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)

PF(
  F,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

F	numeric vector that contains the future value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as a percent
frequency	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

P is expressed as

$$P = F \left[\frac{1}{\left(1 + i\right)^n}\right]$$

P the "present equivalent"

F the "future equivalent"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

PgivenF numeric vector that contains the present value(s) rounded to 2 decimal places

PF data.frame of both n (0 to n) and the resulting present values rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 128, 142, 164.

Examples

```
library(iemisc)

# Example 4-4 from the Reference text (page 128)
PgivenF(10000, 6, 8, "annual") # the interest rate is 8%

PF(10000, 6, 8, "annual") # the interest rate is 8%
```

`PgivenFcont` *Present value given Future value [continuous] (Engineering Economics)*

Description

Compute P given F with interest compounded continuously

Usage

```
PgivenFcont(F, n, r)
```

Arguments

`F`	numeric vector that contains the future value(s)
`n`	numeric vector that contains the period value(s)
`r`	numeric vector that contains the continuously compounded nominal annual interest rate(s) as a percent

Details

P is expressed as

$$P = Fe^{-rn}$$

P the "present equivalent"

F the "future equivalent"

r the "nominal annual interest rate, compounded continuously"

n the "number of periods (years)"

Value

PgivenFcont numeric vector that contains the present value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 169.

Examples

```
library(iemisc)

PgivenFcont(1000, 9, 7) # the interest rate is 7%
```

PgivenFivary *"Present equivalent of a series of future cash flows subject to varying interest rates" (Engineering Economics)*

Description

Compute P given F and i that varies

Usage

```
PgivenFivary(Fn, ik, k)
```

Arguments

Fn	numeric vector that contains the future value(s) at the end of a period n
ik	numeric vector that contains the effective interest rate(s) per period as a percent for the kth period
k	numeric vector that contains the kth period values

Details

P is expressed as

$$P = \frac{F_n}{\prod\limits_{k=1}^{n} (1 + i_k)}$$

P the "present equivalent"

F_n the "future cash flows subject to varying interest rates"

i_k the "interest rate for the kth period"

k the "number of interest periods"

Value

PgivenFivary numeric vector that contains the present value(s)

Author(s)

Irucka Embry

Source

1. r - Add a Column to a Dataframe From a List of Values - Stack Overflow answered by Matthew Plourde on Jun 21 2012. See https://stackoverflow.com/questions/11130037/add-a-column-to-a-datafram 11130178.
2. r - Why does is.vector() return TRUE for list? - Stack Overflow answered by Andrie on May 17 2011. See https://stackoverflow.com/questions/6032772/why-does-is-vector-return-true-for-list/ 6032909.

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 142, 162.

Examples

```
library(iemisc)

# Example for equation 4-31 from the Reference text (page 162)
PgivenFivary(Fn = 1000, ik = c(10, 12, 13, 10), k = 1)
# i1 is 10\%, i2 is 12\%, i3 is 14\%, and i4 is 10\% & k = 1 year
```

PgivenG *Present value given Gradient value (Engineering Economics)*

Description

Compute P given G

Usage

```
PgivenG(
  G,
  n,
  i,
  frequency = c("annual", "semiannual", "quarter", "bimonth", "month", "daily")
)
```

Arguments

G	numeric vector that contains the gradient value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as a percent
frequency	character vector that contains the frequency used to obtain the number of periods [annual (1), semiannual (2), quarter (4), bimonth (6), month (12), daily (365)]

Details

$$P = G \left\{ \frac{1}{i} \left[\frac{(1+i)^n - 1}{i\,(1+i)^n} - \frac{n}{(1+i)^n} \right] \right\}$$

P the "present equivalent"

G the "uniform gradient amount"

i the "effective interest rate per interest period"

n the "number of interest periods"

Value

PgivenG numeric vector that contains the present value(s) rounded to 2 decimal places

Author(s)

Irucka Embry

References

William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 142, 150, 152-154.

Examples

```
library(iemisc)

# Example 4-20 from the Reference text (pages 153-154)
PgivenG(1000, 4, 15, "annual") # the interest rate is 15%
```

---

polygon_area — *Polygon Area (using the Shoelace Formula)*

Description

Calculate the area of a polygon using the shoelace formula.

Usage

```
polygon_area(x, y, plot = c(0, 1), fill = NULL, color = NULL)
```

Arguments

x	numeric vector that contains the x coordinates of the vertices. Regardless of rather the user starts in a clockwise or a counter-clockwise direction, the result will be positive.
y	numeric vector that contains the y coordinates of the vertices. Regardless of rather the user starts in a clockwise or a counter-clockwise direction, the result will be positive.
plot	integer vector that contains 0, 1 only. 0 represents do not plot the polygon and 1 is for plotting the polygon.
fill	character vector that contains the inside color of the polygon. The possible colors are those that are accepted by ggplot2. The default fill color is black.
color	character vector that the border color of the polygon. The possible colors are those that are accepted by ggplot2. The default fill color is black.

Value

the area of the polygon as a positive numeric vector.

Author(s)

John D Page for the JavaScript code, Irucka Embry (R code)

References

John D Page, From Math Open Reference: Algorithm to find the area of a polygon. See https://web.archive.org/web/20221006001150/https://www.mathopenref.com/coordpolygonarea2.html. Provided by Internet Archive: Wayback Machine to avoid the connection timeout.

See Also

polyarea and polyarea

Examples

```
# Example 1 from Source 2

library(iemisc)

x <- c(4,  4,  8,  8, -4, -4)
y <- c(6, -4, -4, -8, -8, 6)

polygon_area(x, y, plot = 1)

# compare with pracma's and geometry's polyarea

pracma::polyarea(x, y)

geometry::polyarea(x, y)




# Example 2

library(iemisc)

type38 <- construction_decimal("46'-10 1/2\"", result = "traditional", output = "vector")

x38 <- c(0, 25, sum(25, type38, 10), sum(25, type38, 10, 25))

y38 <- c(0, rep((3 + 1 / 3), 2), 0)

polygon_area(x38, y38, plot = 1, fill = "darkseagreen3", color = "aquamarine4")

# compare with pracma's and geometry's polyarea

pracma::polyarea(x38, y38)

geometry::polyarea(x38, y38)
```

```
# Example 3

install.load::load_package("iemisc", "data.table")

coords <- fread("
X,   Y
0,0
34,4
58,4
84,6.7
184,0", header = TRUE)

polygon_area(coords$X, coords$Y, plot = 1, color = "#00abff", fill = NA)
# "Use NA for a completely transparent colour." (from ggplot2 color function)

# compare with pracma's and geometry's polyarea

pracma::polyarea(coords$X, coords$Y)

geometry::polyarea(coords$X, coords$Y)




# Example 4 from pracma

library(iemisc)

Xx <- c(0, 4, 4, 0)

Yy <- c(0, 0, 4, 4)

polygon_area(Xx, Yy, 1, color = "goldenrod1", fill = "#00abff")

# compare with pracma's and geometry's polyarea

pracma::polyarea(Xx, Yy)

geometry::polyarea(Xx, Yy)




# Example 5 from pracma

library(iemisc)

Xx1 <- c(0, 4, 2)

Yy1 <- c(0, 0, 4)
```

```
polygon_area(Xx1, Yy1, 1, color = "rosybrown", fill = "papayawhip")

# compare with pracma's and geometry's polyarea

pracma::polyarea(Xx1, Yy1)

geometry::polyarea(Xx1, Yy1)
```

---

project_midpoint *Calculate the midpoint between two coordinates (KY and TN)*

Description

Takes Kentucky or Tennessee-based Northing and Easting engineering survey measurements [based in the State Plane Coordinate System (SPCS)] in meters, international foot, or US survey foot and converts those values into geodetic coordinates of the World Geodetic System (WGS) (19)84 (EPSG:4326). [MapTiler Reference] Each latitude [Y] and longitude [X] point is verified to be located within Kentucky or Tennessee.

Usage

```
project_midpoint(
  Northing_begin,
  Easting_begin,
  Northing_end,
  Easting_end,
  units = c("survey_ft", "foot", "meters"),
  location = c("KY", "TN"),
  output = c("simple", "advanced")
)
```

Arguments

Northing_begin	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Northing engineering survey measurement in meters, international foot, or US survey foot
Easting_begin	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Northing engineering survey measurement in meters, international foot, or US survey foot
Northing_end	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Northing engineering survey measurement in meters, international foot, or US survey foot

Easting_end	numeric vector (or character vector with numbers, commas, and decimal points) that contains the Northing engineering survey measurement in meters, international foot, or US survey foot
units	character vector that contains the system of units (options are survey_ft (United States Customary System) [US survey foot], foot, or meters (International System of Units) [meters] [only 1 set of units at a time])
location	character vector that contains the location name ('KY' for Kentucky or 'TN' for Tennessee) [only 1 location at a time]
output	character vector that contains simple for the default result using a simple data.table or advanced for the result as a complex data.table

Value

the projected associated latitude [Y] and longitude [X] mid point coordinates in Decimal Degrees as a data.table or as an enhanced data.table with the Northing and Easting coordinates in US survey foot, international foot, and meters in addition to the [Y] and [X] coordinates for the begin, middle, and end points

Note

Please Note: If you have Kentucky North/South Zone survey measurements, then please use the Kentucky Geological Survey, University of Kentucky - Kentucky Single Coordinate Conversion Tool (http://kgs.uky.edu/kgsweb/CoordConversionTool.asp) instead. That tool will give you the geographic coordinates too. This R function, project_midpoint will only be valid for NAD83 / Kentucky Single Zone.

Useful Tennessee reference Web site Tennessee Department of Transportation Roadway Design Survey Standards https://www.tn.gov/tdot/roadway-design/survey-standards.html

Useful Kentucky reference Web site Kentucky Transportation Cabinet Survey Coordination https://transportation.ky.gov/Highway-Design/Pages/Survey-Coordination.aspx

Author(s)

Irucka Embry

Source

1. Win-Vector Blog. John Mount, June 11, 2018, "R Tip: use isTRUE()", https://win-vector.com/2018/06/11/r-tip-use-istrue/.
2. Latitude Longitude Coordinates to State Code in R - Stack Overflow answered by Josh O'Brien on Jan 6 2012 and edited by Josh O'Brien on Jun 18, 2020. See https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r.
3. r - Convert column classes in data.table - Stack Overflow answered by Matt Dowle on Dec 27 2013. See https://stackoverflow.com/questions/7813578/convert-column-classes-in-data-table.
4. Excel vlook up function in R for data frame - Stack Overflow answered by Tyler Rinker on Apr 8 2013 and edited by Tyler Rinker on Feb 26 2014. See https://stackoverflow.com/questions/15882743/excel-vlook-up-function-in-r-for-data-frame.

5. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
6. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-askec

References

1. udunits.dat, v 1.18 2006/09/20 18:59:18 steve Exp, https://web.archive.org/web/20230202155021/https://www.unidata.ucar.edu/software/udunits/udunits-1/udunits.txt. Retrieved thanks to the Internet Archive: Wayback Machine
2. Spatial Reference, Aug. 13, 2004, "EPSG:3088: NAD83 / Kentucky Single Zone", https://spatialreference.org/ref/epsg/3088/.
3. Spatial Reference, March 7, 2000, "EPSG:32136 NAD83 / Tennessee", https://spatialreference.org/ref/epsg/32136/.
4. MapTiler Team, "EPSG:4326: WGS 84 – WGS84 - World Geodetic System 1984, used in GPS, https://epsg.io/4326.
5. Tennessee Department of Transportation Design Division, Tennessee Department of Transportation Tennessee Geodetic Reference Network (TGRN) Reference Manual Second Edition Issued, page ix, https://www.tn.gov/content/dam/tn/tdot/documents/TgrnComposite.pdf.
6. Earth Point, "State Plane Coordinate System - Convert, View on Google Earth", https://www.earthpoint.us/StatePlane.aspx.
7. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, Mid Valley Oil Rad Relay Twr designation, HA1363 PID, Grayson County Kentucky, Clarkson (1967) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=HA1363.
8. National Geodetic Survey datasheet95, version 8.12.5.3, online retrieval date July 25, 2019, 2006 42 07 designation, DL4005 PID, Fayette County Kentucky, Lexington West (1993) USGS Quad, https://www.ngs.noaa.gov/cgi-bin/ds_mark.prl?PidBox=DL4005.

Examples

```
# Example 1

library(iemisc)

Northing_begin <- 283715.8495
Easting_begin <- 1292428.3999

Northing_end <- 303340.6977
Easting_end <- 1295973.7743

project_midpoint(Northing_begin, Easting_begin, Northing_end, Easting_end,
units = "survey_ft", location = "TN", output = "simple")
```

```
# See Source 5 and Source 6

# Please see the error messages

library(iemisc)

# Tennessee (TN) Northing and Easting in meters

Northing2 <- c(232489.480, 234732.431)

Easting2 <- c(942754.124, 903795.239)

dt4 <- try(project_midpoint(Northing2, Easting2, units = "survey_ft",
location = "TN", output = "simple"))
```

---

prop_mortality_ratio *Proportional Mortality Ratio*

Description

This function calculates the proportional mortality ratio for a given population.

Usage

```
prop_mortality_ratio(cause_deaths, total_deaths)
```

Arguments

cause_deaths	numeric vector that contains the deaths from a single cause
total_deaths	numeric vector that contains the total deaths in the given population

Value

the proportional mortality ratio (as a percent) as a numeric vector

Author(s)

Irucka Embry

References

Michael Darcy and Łucja Zaborowska, MD, PhD, Last updated on Nov 05, 2022, "Mortality Rate Calculator", Omni Calculator, https://www.omnicalculator.com/health/mortality-rate. Florida Museum of Natural History: International Shark Attack File, Last updated on 07/19/2022, "Risk of Death: 18 Things More Likely to Kill You Than Sharks", https://www.floridamuseum.ufl.edu/shark-attacks/odds/compare-risk/death/. Farida B. Ahmad, MPH, Jodi A. Cisewski, MPH, Robert N. Anderson, PhD, *MMWR Morb Mortal Wkly Rep* 2022, 71:597-600, "Provisional Mortality Data — United States, 2021", https://www.cdc.gov/mmwr/volumes/71/wr/mm7117e1.htm.

Examples

```
# Data from Reference 2 and Reference 3

library(iemisc)

prop_mortality_ratio(cause_deaths = 652486, total_deaths = 3458697)
# annual heart disease deaths & total deaths in the US in 2021
```

---

prop_solver *Proportion Solver*

Description

Solve the proportion for the missing numeric value in either of the numerators or denominators

Usage

```
prop_solver(
  n1 = NULL,
  d1 = NULL,
  n2 = NULL,
  d2 = NULL,
  output = c("single", "all")
)
```

Arguments

n1	numeric vector that contains the numerator 1
d1	numeric vector that contains the denominator 1
n2	numeric vector that contains the numerator 2
d2	numeric vector that contains the denominator 2
output	character vector that contains the output type, (all or single)

Value

the missing proportion value as either a single numeric value or all values as characters

Author(s)

Irucka Embry

Source

1. Basic Mathematics. "Solving Proportions", https://www.basic-mathematics.com/solving-proportions.html
2. Dr. Ariyana Love. "SMasks And Covid Tests Contain Nanotech Vaccines Without Informed Consent", April 7, 2021, https://ambassadorlove.blog/2021/04/07/masks-and-covid-tests-contain-nanote
3. "Electronic Support for Public Health–Vaccine Adverse Event Reporting System (ESP:VAERS) Grant Final Report"/Grant ID: R18 HS 017045 [Inclusive dates: 12/01/07 - 09/30/10]. Principal Investigator: Lazarus, Ross, MBBS, MPH, MMed, GDCompSci., page 6, https://web.archive.org/web/20211230233658/https://www.nvic.org/CMSTemplates/NVIC/Pdf/FDA/ahrq-vaers-report-2011.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Example 1 from the Example # 1 from Reference 1

library(iemisc)

# 5 / x = 10 / 16

prop_solver(n1 = 5, n2 = 10, d2 = 16, output = "single")




# Example 2

library(iemisc)

t1 <- "34 3 1/2"
t2 <- 5
t3 <- 5 / 2

t11 <- construction_decimal(t1, result = "traditional", output = "vector")

prop_solver(n1 = t11, n2 = 5, d1 = 5 / 2, output = "all")




# Example 3
```

```
library(iemisc)

# Refer to Reference 2 and Reference 3

# What is the numerator (n1) for the situation where VAERS reports 4,576 dead
# people; however, the number of dead people is closer to 453,024 people?

d1 <- 100 / 100 # 100%
n2 <- 4576 # number of deceased people
d2 <- 453024 # number of deceased people

prop_solver(d1 = d1, n2 = n2, d2 = d2, output = "single")


# What is the more accurate number of dead people (d2) where VAERS reports 4,576
# dead people and we recognize that less than 1% of adverse reactions are reported
# to VAERS?

n1 <- 0.99999999999999999999999999999999999999999999999 / 100 # less than 1%
n11 <- 0.98 / 100 # less than 1%
d1 <- 100 / 100 # number of deceased people
n2 <- 4576 # number of deceased people

prop_solver(n1 = n1, d1 = d1, n2 = n2, output = "all")

prop_solver(n1 = n1, d1 = d1, n2 = n2, output = "single")


prop_solver(n1 = n11, d1 = d1, n2 = n2, output = "all")

prop_solver(n1 = n11, d1 = d1, n2 = n2, output = "single")
```

`rain_garden_driveway` *Rain Garden Sizing for Driveways*

Description

This function computes the final rain garden dimensions based on the size of the impervious roof surfaces and the initial rain garden size. This function uses the `surface_area` values and units in the calculations.

Usage

```
rain_garden_driveway(
  dw_length = NULL,
  dw_width = NULL,
  rf_length = NULL,
  rf_width = NULL,
  driveway_table = NULL,
  roof_table = NULL,
  lw_units = c("inch", "feet", "survey_foot", "yard", "mile", "centimeter", "meter",
    "kilometer"),
  rainfall_depth,
  rainfall_depth_units = c("inch", "feet", "centimeter", "meter"),
  rain_garden_depth,
  rain_garden_depth_units = c("inch", "feet", "centimeter", "meter")
)
```

Arguments

`dw_length`	numeric vector containing the length value(s) in one of the lw_units values for the driveway(s).
`dw_width`	numeric vector containing the width value(s) in one of the lw_units values for the driveway(s).
`rf_length`	numeric vector containing the length value(s) in one of the lw_units values for the roof(s).
`rf_width`	numeric vector containing the width value(s) in one of the lw_units values for the roof(s).
`driveway_table`	data.frame/data.table/tibble, list, or matrix containing the length in column 1 and the width in column 2 for the driveway(s).
`roof_table`	data.frame/data.table/tibble, list, or matrix containing the length in column 1 and the width in column 2 for the roof(s).
`lw_units`	character vector containing the units for the length and the width (default = "feet"). The other possible units are "inch", "survey_foot", "yard", "mile", "centimeter", "meter", or "kilometer". The units should be consistent and not mixed.
`rainfall_depth`	numeric vector containing the rainfall depth in one of the rainfall_depth_units.
`rainfall_depth_units`	character vector containing the units for the rainfall depth (default = "feet"). The other possible units are "inch", "centimeter", or "meter".
`rain_garden_depth`	numeric vector containing the rain garden depth in one of the rain_garden_depth_units.
`rain_garden_depth_units`	character vector containing the units for the rain garden depth (default = "feet"). The other possible units are "inch", "centimeter", or "meter".

Value

a data.table with the following columns: Driveway Drainage Area, One-Quarter of Roof Area, Depth of Rain, Design Storm Volume, Rain Garden Depth, Rain Garden Initial Size, Rain Garden Initial Dimensions, Total Drainage Area, New Design Storm Volume, Rain Garden Final Size, Rain Garden Final Dimensions

Author(s)

Irucka Embry

References

Green Infrastructure Champion Training: Part 7: "How To Design and Build a Rain Garden", April 10, 2019, pages 41 - 43 of the PDF document, https://water.rutgers.edu/Projects/GreenInfrastructureChampions/Talks_2020/Part_7_04102020.pdf.

See Also

surface_area for calculating the linear surface area

Examples

```
# Note: the units must be consistent for the lengths and widths

# Example 1 (from the Reference)

library(iemisc)

dw_width1 <- c(15, 10)
dw_length1 <- c(50, 25)
lw_units <- "feet"
rf_width1 <- 50
rf_length1 <- 25
rainfall_depth1 <- 1.5
rainfall_depth_units <- "inch"
rain_garden_depth <- 6
rain_garden_depth_units <- "inch"

rain_garden_driveway(dw_length = dw_length1, dw_width = dw_width1, rf_length =
rf_length1, rf_width = rf_width1, lw_units = lw_units, rainfall_depth =
rainfall_depth1, rainfall_depth_units = rainfall_depth_units, rain_garden_depth
= rain_garden_depth, rain_garden_depth_units = rain_garden_depth_units)



# Example 2
# from https://www.ecoccs.com/R_Examples/Simple-Rain-Garden-Sizing_with-R.html
# Irucka Embry modified the Example from the Reference for this example

install.load::load_package("iemisc", "data.table")
```

```
dw_length2 <- c(construction_decimal("50 feet 3 1/2 inch", result <- "traditional",
output <- "vector"), construction_decimal("25 feet 5 7/8 inch", result <- "traditional",
output <- "vector"))
dw_width2 <- c(construction_decimal("15 feet 10 3/4 inch", result <- "traditional",
output <- "vector"), construction_decimal("10 feet 7 3/8 inch", result <- "traditional",
output <- "vector"))
lw_units <- "feet"
rf_length2 <- construction_decimal("25 feet 10 1/4 inch", result <- "traditional",
output <- "vector")
rf_width2 <- construction_decimal("12.5 feet 1 1/8 inch", result <- "traditional",
output <- "vector") * 4
rainfall_depth2 <- 2.25
rainfall_depth_units <- "inch"
rain_garden_depth <- 6
rain_garden_depth_units <- "inch"

driveway_table <- data.table(length = dw_length2, width = dw_width2)

roof_table <- data.table(length = rf_length2, width = rf_width2)

rain_garden_driveway(driveway_table = driveway_table, roof_table = roof_table,
lw_units = lw_units, rainfall_depth = rainfall_depth2, rainfall_depth_units =
rainfall_depth_units, rain_garden_depth = rain_garden_depth,
rain_garden_depth_units = rain_garden_depth_units)
```

`ranges` *Sample range*

Description

This function computes the sample range.

Usage

```
ranges(x, na.rm = FALSE, finite = FALSE)
```

Arguments

`x`	any numeric vector
`na.rm`	logical vector that determines whether the missing values should be removed or not. The default is FALSE.
`finite`	logical vector that determines whether non-finite values should be removed or not. The default is FALSE.

Details

"The range is the difference between the largest number and the smallest number in the set." Source: Onwubiko page 176.

The following statements are from range:

"If na.rm is FALSE, NA and NaN values in any of the arguments will cause NA values to be returned, otherwise NA values are ignored."

"If finite is TRUE, the minimum and maximum of all finite values is computed, i.e., finite = TRUE includes na.rm = TRUE."

Value

ranges as the difference between the maximum and minimum values in x as a numeric vector. Unlike the range, ranges can't take character vectors as arguments, only numeric vectors.

Author(s)

Irucka Embry

Source

1. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
2. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-asked

References

Chinyere Onwubiko, *An Introduction to Engineering*, Mission, Kansas: Schroff Development Corporation, 1997, page 176.

See Also

sgm for geometric mean, shm for harmonic mean, cv for coefficient of variation (CV), rms for root-mean-square (RMS), relerror for relative error, and approxerror for approximate error.

Examples

```
# Example 1

install.load::load_package("iemisc", "rando")

set_n(100) # makes the example reproducible

x <- r_norm(.seed = 943)

ranges(x)
```

```
install.load::load_package("iemisc", "rando")

set_n(100) # makes the example reproducible

(r.x <- ranges(r_norm(.seed = 100))); r.x


# See Source 1 and Source 2

# Example 2 (from the base range function)

library(iemisc)

xi <- c(NA, 1:3, -1:1/0); xi

try(ranges(xi))

try(ranges(xi, na.rm = TRUE))

try(ranges(xi, finite = TRUE))
```

---

rational_formula *Modified Rational Method Equation*

Description

Computes the design peak runoff rate (Q) using the modified rational method equation.

Usage

```
rational_formula(
  C_F,
  C,
  i,
  A,
  area_units = c("acre", "square feet", "square mile", "hectare", "square kilometer")
)
```

Arguments

C_F numeric vector that contains the "runoff coefficient adjustment factor to account for reduction of infiltration and other losses during high intensity storms" [Input a number between 2 and 10; 25; 50; or 100]

`C`	numeric vector that contains the dimensionless "runoff coefficient to reflect the ratio of rainfall to surface runoff"
`i`	numeric vector that contains the "rainfall intensity in inches per hour (in/hr)"
`A`	numeric vector that contains the drainage area in one of the area_units values.
`area_units`	character vector containing the units for area (default = "acre"). The other possible units are "square feet", "square mile", "hectare", or "square kilometer".

Value

the numeric vector Q, which is the "peak flow" in cubic feet per second (cfs or ft^3/s)

Note

Please note: Refer to the limitations of the Modified Rational Method equation for your particular jurisdiction. Notes are only included below for Oklahoma and Oregon, respectively.

for Oklahoma "The Rational Method, first introduced in 1889, is recommended for estimating the design storm peak runoff for areas up to 640 acres. The Rational Method was modified in the 1980's to include a runoff coefficient correction tied to the flood frequency. This Modified Rational Method is used by ODOT.

Some precautions should be considered when applying the Rational method: –The first step in applying the Rational method is to obtain a good topographic map and define the boundaries of the drainage area in question. A field inspection of the area should also be made to determine if the natural drainage divides have been altered. –In determining surface characteristics for the drainage area, consider any future changes in land use that might occur during the service life of the proposed facility that could result in an inadequate drainage system. Also, the effects of upstream detention facilities may be considered. –Restrictions to the natural flow (e.g., highway crossings and dams that exist in the drainage area) should be investigated to determine how they might affect the design flows. –The charts, graphs and tables included in this Section are not intended to replace reasonable and prudent engineering judgment that should permeate each step in the design process." [Oklahoma Department of Transportation Reference]

for Oregon: "Limitations and assumptions in the Rational Method are as follows: –The drainage area should not be larger than 200 acres. –The peak flow is assumed to occur when the entire watershed is contributing runoff. –The rainfall intensity is assumed to be uniform over a time duration equal to or greater than the time of concentration, Tc. –The peak flow recurrence interval is assumed to be equal to the rainfall intensity recurrence interval. In other words, the 10-year rainfall intensity is assumed to produce the 10-year flood." [Oregon Department of Transportation Reference]

The value of 1.008 is used for the unit conversion factor for English units. [Tennessee Design reference]

Author(s)

Irucka Embry

References

1. Design Principles for Erosion Prevention & Sediment Control for Construction Sites Level II EPSC Workshop, Fall 2017. Sponsored by The University of Tennessee Biosystems Engineering & Environmental Sciences Tennessee Water Resources Research Center, Tennessee Department of Environment and Conservation Division of Water Resources, and Tennessee Department of Transportation.
2. Oklahoma Department of Transportation (ODOT) Roadway Drainage Manual Chapter 7 Hydrology, November 2014, page 7.6-1, https://oklahoma.gov/content/dam/ok/en/odot/documents/chapter-7-hydrology.pdf.
3. Oregon Department of Transportation (ODOT) Geo-Environmental, ODOT Hydraulics Manual Appendix F – Rational Method, April 2014, page 7-F-1, https://web.archive.org/web/20221202194502/https://www.oregon.gov/odot/GeoEnvironmental/Docs_Hydraulics_Manual/Hydraulics-07-F.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
4. U.S. Department of Agriculture (USDA) Natural Resources Conservation Service (NRCS), Hydrology Training Series Module 206 D - Peak Discharge (Other Methods) Study Guide, page 18 (of the PDF document) and page 26 - 27 (of the PDF document), https://web.archive.org/web/20211018222532/https://www.nrcs.usda.gov/Internet/FSE_DOCUMENTS/stelprdb1083019.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Example 1 from NRCS Reference

# Given
# Urban setting with a drainage area of 12 acres
# 6 acres = single family area
# 3 acres = park
# 3 acres = streets (concrete)
# Soil = clay loam
# Tc = 20 min (time of concentration)

# Find the instantaneous peak discharge for a 25-yr frequency flood at a
# road crossing in an urban/rural area located in the Kansas City, Missouri
# area.


library(iemisc)

area1 <- c(6, 3, 3)
C1 <- c(mean(c(0.30, 0.50)), 0.15, 0.90)
C1_w <- weighted_C(C = C1, area = area1)

i1 <- 5.1 # in/hr

rational_formula(C_F = 25, C = C1_w, i = i1, A = sum(area1), area_units = "acre")
```

```
# Example 2 from NRCS Reference

# Given
# Urban setting with a drainage area of 18 acres

# 1 ac = playground
# 10 ac = single family area
# 2 ac = streets (asphaltic)
# 5 ac = pasture (hilly)
# Soil = heavy clay
# Tc = 20 min

# Find the instantaneous 100-yr frequency peak discharge for design of a
# channel in a developing subdivision located in an area near Asheville,
# North Carolina.


library(iemisc)

area2 <- c(1, 10, 2, 5)
C2 <- c(0.35, 0.50, 0.90, 0.60)
C2_w <- weighted_C(C = C2, area = area2)

i2 <- 5.5 # in/hr

rational_formula(C_F = 100, C = C2_w, i = i2, A = sum(area2), area_units = "acre")
```

Re1 *Calculating the Reynolds Number 1*

Description

Various Methods of Calculating the Reynolds number

Usage

```
Re1(D, V, rho, mu, gc = NULL, units = c("SI", "Eng", "slug"))
```

Arguments

- `D` numeric vector that contains the hydraulic diameters '(four times the area in flow divided by the wetted surface) is a characteristic length' (m or ft) [Reference: Lindeburg Manual]
- `V` numeric vector that contains the average fluid velocity (m/s or ft/s) [Reference: Lindeburg Manual]
- `rho` numeric vector that contains the fluid density (kg/m^3 or lbm/ft^3) [Reference: Lindeburg Manual]
- `mu` numeric vector that contains the absolute or dynamic viscosity of the fluid (Pa-s or lbf-sec/ft^2) [Reference: Lindeburg Manual]
- `gc` numeric vector that contains the gravitational constant (32.2 lbm-ft/lbf-sec^2) [Reference: Lindeburg Manual]
- `units` character vector that contains the system of units [options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), and `slug` for the English unit slug which is a consistent unit of mass]

Details

The Reynolds number, named after Osborne Reynolds, is a dimensionless number that is used to determine the type of fluid flow (laminar, transition, or turbulent). [References: Lindeburg Manual and Subramanian]

Re1 - uses the absolute or dynamic viscosity (\mu) Re2 and Re4 - use kinematic viscosity (\nu) Re3 - uses the 'mass flow rate per unit area' (G) [Reference: Lindeburg Manual]

The Reynolds number equation can be expressed in the following ways [Reference: Lindeburg Manual]:

$$Re = \frac{inertial_forces}{viscous_forces}$$

$$Re = \frac{DVrho}{mu}$$

Re the Reynolds number (dimensionless)

D the hydraulic diameters (m)

V average velocity of the fluid (m/s)

rho density of the fluid at a certain temperature (kg/m^3)

mu absolute or dynamic viscosity of the fluid at a certain temperature (Pa-s)

$$Re = \frac{DVrho}{mug_c}$$

Re the Reynolds number (dimensionless)

D the hydraulic diameters (ft)

V average velocity of the fluid (ft/sec)

\rho density of the fluid at a certain temperature (lbm/ft^3)

\mu absolute or dynamic viscosity of the fluid at a certain temperature (lbf-sec/ft^2)

g_c gravitational constant (32.2 lbm-ft/lbf-sec^2) used for dimensional analysis so that the Reynolds number will be dimensionless with US Customary units

$$Re = \frac{DG}{mu}$$

Re the Reynolds number (dimensionless)

D the hydraulic diameters (m)

G 'mass flow rate per unit area' (kg/m^2-s)

\mu absolute or dynamic viscosity of the fluid at a certain temperature (Pa-s)

$$Re = \frac{DG}{g_c mu}$$

Re the Reynolds number (dimensionless)

D the hydraulic diameters (ft)

G 'mass flow rate per unit area' (lbm/ft^2-sec)

\mu absolute or dynamic viscosity of the fluid at a certain temperature (lbf-sec/ft^2)

g_c gravitational constant (32.2 lbm-ft/lbf-sec^2) used for dimensional analysis so that the Reynolds number will be dimensionless with US Customary units

where

$$G = rhoV$$

G 'mass flow rate per unit area' (kg/m^2-s)

\rho density of the fluid at a certain temperature (kg/m^3)

V average velocity of the fluid (m/s)

$$Re = \frac{DV}{nu}$$

Re the Reynolds number (dimensionless)

D the hydraulic diameters (m)

V average velocity of the fluid (m/s)

\nu kinematic viscosity of the fluid at a certain temperature (m^2/s)

$$Re = \frac{DV}{nu g_c}$$

Re the Reynolds number (dimensionless)

D the hydraulic diameters (ft)

V average velocity of the fluid (ft/sec)

\nu absolute or dynamic viscosity of the fluid at a certain temperature (lbf-sec/ft^2)

g_c gravitational constant (32.2 lbm-ft/lbf-sec^2) used for dimensional analysis so that the Reynolds number will be dimensionless with US Customary units

where

$$nu = \frac{mu}{rho}$$

\nu kinematic viscosity of the fluid at a certain temperature (m^2/s)

\mu absolute or dynamic viscosity of the fluid at a certain temperature (Pa-s)

\rho density of the fluid at a certain temperature (kg/m^3)

where

$$nu = \frac{mu g_c}{rho}$$

\nu kinematic viscosity of the fluid at a certain temperature (ft^2/sec)

\mu absolute or dynamic viscosity of the fluid at a certain temperature (lbf-sec/ft^2)

g_c gravitational constant (32.2 lbm-ft/lbf-sec^2) used for dimensional analysis so that the kinematic viscosity units will work in US Customary units

\rho density of the fluid at a certain temperature (lbm/ft^3)

Value

the Reynolds number as a `list` for Re1

Note

Please Note: The conventional wisdom that a Reynolds number less than 2100 is laminar flow, between 2100 and 4000 is transitional or critical flow, and greater than 4000 is turbulent flow is not accurate. 'Reynolds himself observed that turbulence was triggered by inlet disturbances to the pipe and the laminar state could be maintained to Re \u2248 12,000 if he took great care in minimizing external disturbances to the flow. By careful design of pipe entrances Ekman (1910) has maintained laminar pipe flow up to a Reynolds number of 40,000 and Pfenniger (1961) up to 100,000 by minimising ambient disturbances.' [References: Lindeburg Manual and Trinh]

'Numerous experiments have shown that the flow in a pipe changes from laminar to turbulent in the range of R between the critical value of 2,000 and a value that may be as high as 50,000.* In these experiments the diameters of the pipe was taken as the characteristic length in defining the Reynolds number. When the hydraulic radius is taken as the characteristic length, the corresponding range is from 500 to 12,500,* since the diameters of a pipe is four times its hydraulic radius. * = It should be noted that there is actually no definite upper limit.' [Reference: Chow]

Note: Units must be consistent

Author(s)

Irucka Embry

References

1. Ven Te Chow, Ph.D., *Open-Channel Hydraulics*, McGraw-Hill Classic Textbook Reissue, New York City, New York: McGraw-Hill Book Company, 1988, pages 7-8.
2. Michael R. Lindeburg, PE, *Civil Engineering Reference Manual for the PE Exam*, Twelfth Edition, Belmont, California: Professional Publications, Inc., 2011, pages 17-1, 17-5, 17-8 - 17-9.
3. The NIST Reference on Constants, Units, and Uncertainty, Fundamental Constants Data Center of the NIST Physical Measurement Laboratory, "standard acceleration of gravity g_n", https://web.archive.org/web/20230427133623/https://physics.nist.gov/cgi-bin/cuu/Value?gn. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.
4. R. Shankar Subramanian, "Pipe Flow Calculations", page 9, Clarkson University Department of Chemical and Biomolecular Engineering, https://web2.clarkson.edu/projects/subramanian/ch330/notes/Pipe%20Flow%20Calculations.pdf.
5. R. Shankar Subramanian, "Reynolds Number", page 1, Clarkson University Department of Chemical and Biomolecular Engineering, https://web2.clarkson.edu/projects/subramanian/ch330/notes/Reynolds%20Number.pdf.
6. Khanh Tuoc Trinh, "On the Critical Reynolds Number for Transition From Laminar to Turbulent Flow", page 2, https://arxiv.org/abs/1007.0810.
7. Wikimedia Foundation, Inc. Wikipedia, 15 May 2019, "Conversion of units", https://en.wikipedia.org/wiki/Conversion_of_units.

See Also

f1, f2, f3, f4, f5, f6, f7, and f8 for the Darcy friction factor (f) for pipes

Re2, Re3, Re4

Examples

```
# from Lindeburg Reference page 17-8
# D = 0.3355 ft
# V = 7.56 ft/sec

# from the Chow reference, water at 68 F (20 C) has the following properties

library(iemisc)

# mu (dynamic viscosity) = 2.09 * 10 ^ -5 slug/ft-sec
# rho (density) = 1.937 slug/ft^3
# v (kinematic viscosity) = mu / rho = 1.08 * 10 ^ -5

Re1(D = 0.3355, V = 7.56, rho = 1.937, mu = 2.09 * 10 ^ -5, units = "slug")
```

Re2 *Calculating the Reynolds Number 2*

Description

Calculating the Reynolds Number 2

Usage

```
Re2(D, V, nu)
```

Arguments

D	numeric vector that contains the hydraulic diameters "(four times the area in flow divided by the wetted surface) is a characteristic length" (m or ft) [Reference: Lindeburg Manual]
V	numeric vector that contains the average fluid velocity (m/s or ft/s) [Reference: Lindeburg Manual]
nu	numeric vector that contains the kinematic viscosity of the fluid (m^2/s or lbf-sec/ft^2) [Reference: Lindeburg Manual]

Value

the Reynolds number as a numeric vector for Re2

Author(s)

Irucka Embry

See Also

Re1 for the additional seealso, description, details, note, and references sections, Re3, Re4

Examples

```
# from Lindeburg Reference page 17-8
# where
# D = 0.3355 ft
# V = 7.56 ft/sec
# nu = 1.41 * 10 ^ -5 ft^2 / sec
# and
# Re = 1.8 * 10 ^ 5

library(iemisc)

Re2(D = 0.3355, V = 7.56, nu = 1.41 * 10 ^ -5)

# compare to Re1(D = 0.3355, V = 7.56, rho = 1.937, mu = 2.09 * 10 ^ -5, units = "slug")
```

```
Re2(D = 0.3355, V = 7.56, nu = 1.08 * 10 ^ -5)
```

Re3 *Calculating the Reynolds Number 3*

Description

Calculating the Reynolds Number 3

Usage

```
Re3(D, G, mu, gc = NULL, units = c("SI", "Eng"))
```

Arguments

`D`	numeric vector that contains the hydraulic diameters "(four times the area in flow divided by the wetted surface) is a characteristic length" (m or ft) [Reference: Lindeburg Manual]
`G`	numeric vector that contains the 'mass flow rate per unit area' (kg/m^2-s or lbm/ft^2-sec) [Reference: Lindeburg Manual]
`mu`	numeric vector that contains the absolute or dynamic viscosity of the fluid (Pa-s or lbf-sec/ft^2) [Reference: Lindeburg Manual]
`gc`	numeric vector that contains the gravitational constant (32.2 lbm-ft/lbf-sec^2) [Reference: Lindeburg Manual]
`units`	character vector that contains the system of units [options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), and `slug` for the English unit slug which is a consistent unit of mass]

Value

the Reynolds number as a numeric `vector` for Re3

Note

Note: Please see the Calculating the Reynolds number Examples vignette for usage of Re3

Author(s)

Irucka Embry

See Also

`Re1` for the additional seealso, description, details, note, and references sections, `Re2`, `Re4`

Re4 *Calculating the Reynolds Number 4*

Description

Calculating the Reynolds Number 4

Usage

```
Re4(Q, nu, D)
```

Arguments

Q	numeric vector that contains the discharge value of the fluid (m^3/s or ft^3/s) [Reference: Lindeburg Manual]
nu	numeric vector that contains the kinematic viscosity of the fluid (m^2/s or lbf-sec/ft^2) [Reference: Lindeburg Manual]
D	numeric vector that contains the hydraulic diameters "(four times the area in flow divided by the wetted surface) is a characteristic length" (m or ft) [Reference: Lindeburg Manual]

Value

the Reynolds number as a numeric `vector` for Re4

Author(s)

Irucka Embry

See Also

`Re1` for the additional seealso, description, details, note, and references sections, `Re2`, `Re3`

Examples

```
# part of Example 3 from Subramanian Pipe Flow Calculations

# Q = 2.23 * 10 ^ - 2 ft^3/s
# nu = 2.40 * 10 ^ -5 ft^2/s
# D = 9.03 * 10 ^ -2 ft

library(iemisc)

Re4(Q = 2.23 * 10 ^ -2, nu = 2.40 * 10 ^ -5, D = 9.03 * 10 ^ -2)
```

reduce_single_digit — *Reduce an Integer, a Date (Time), or a Number (with or without Decimals) to a Single Integer*

Description

Takes a character vector coercible to a date using anydate or a date time using anytime; a character vector with numbers; a numeric vector; or an integer vector & computes the sum to a single digit using Mod_octave

The vectors may include periods, dashes, parentheses, colons, and/or spaces. See the examples.

Usage

```
reduce_single_digit(string)
```

Arguments

string	character vector coercible to a date using anytime or a date time using anytime; a numeric vector; or an integer vector

Value

a numeric vector with a single digit (integer from 0 - 9)

Author(s)

Irucka Embry

References

1. Numerology.com, "Number 9 Meaning", https://www.numerology.com/articles/about-numerology/single-digit-number-9-meaning/.
2. Numerology.com, "Numerology Numbers 1-9: Exploring the single digit numbers in Numerology", https://www.numerology.com/articles/about-numerology/single-digit-numbers-in-numerolo
3. GeeksforGeeks, Last updated on 13 Jun, 2022, "Finding sum of digits of a number until sum becomes single digit", https://www.geeksforgeeks.org/finding-sum-of-digits-of-a-number-until-sum-be
4. Wikimedia Foundation, Inc. Wikipedia, 18 November 2022, "Digital root", https://en.wikipedia.org/wiki/Digital_root.

Examples

```
# Please refer to the iemisc: Sound Frequencies & Nikola Tesla's 3-6-9 Theory
# vignette
# https://www.ecoccs.com/R_Examples/SoundFrequencies-and-3-6-9.pdf for
# additional examples
```

```
# Examples

library(iemisc)

reduce_single_digit(37)

reduce_single_digit(5094322.439344993211394)

reduce_single_digit(-438443.349435493)

reduce_single_digit("-48373744582.47362287482374")

reduce_single_digit("11-09-2022")

reduce_single_digit("2001/01/31")

reduce_single_digit("24 December 1983 04:37:58.55543333")

reduce_single_digit("4 July 1776")

reduce_single_digit(9)

reduce_single_digit(0)

reduce_single_digit(94321155)

reduce_single_digit("011 (704) 904-0432")

reduce_single_digit("011-894-908-0945")

reduce_single_digit("908-0945")


datess <- seq(as.Date("2001/07/17"), as.Date("2001/08/03"), by = "day")
datess
xt <- sapply(datess, reduce_single_digit)
xt
datess[which(xt == 3 | xt == 6 | xt == 9)]
```

---

relerror *Relative error*

Description

This function computes the relative error.

Usage

```
relerror(xt, xa)
```

Arguments

`xt`	numeric vector that contains the true value(s)
`xa`	numeric vector that contains the approximate value(s)

Details

Relative error is expressed as

$$\varepsilon_t = \frac{true\ value - approximation}{true\ value} \cdot 100$$

ε_t the "true percent relative error"

true value the true value

approximation the approximate value

Value

relative error, as a percent (%), as a numeric `vector`.

Author(s)

Irucka Embry

References

Steven C. Chapra, *Applied Numerical Methods with MATLAB for Engineers and Scientists*, Second Edition, Boston, Massachusetts: McGraw-Hill, 2008, page 82-83.

See Also

`sgm` for geometric mean, `shm` for harmonic mean, `cv` for coefficient of variation (CV), `rms` for root-mean-square (RMS), `approxerror` for approximate error, and `ranges` for sample range.

Examples

```
library(iemisc)

# Example 4.1 from the Reference text (page 83)

relerror(1.648721, 1.5) # answer as a percent (\%)
```

Rem *Remainder After Division (GNU Octave/MATLAB compatible)*

Description

Obtain the remainder after division in a manner compatible with GNU Octave/MATLAB.

Usage

```
Rem(x, n)
```

Arguments

x, n	An R object (array, matrix, vector)

Value

"Return the remainder of the division x / y." Source: Eaton.

Author(s)

Irucka Embry, Samit Basu (FreeMat)

References

1. Samit Basu (2002-2006). FreeMat v4.0, https://freemat.sourceforge.net/help/mathfunctions_rem.html.
2. John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 564.

Examples

```
# Example from GNU Octave

library(iemisc)

x <- 23.4
y <- 20

Rem(x, y)


# Please refer to the iemisc: Examples from GNU Octave Rem, Mod, and
# fractdiff Compatible Functions vignette for additional examples
```

righttri — *Right triangle calculations*

Description

This function computes the missing length (must have at least 2 sides) and the interior angles (degrees) of a right triangle.

Usage

```
righttri(a = NULL, b = NULL, c = NULL)
```

Arguments

a	numeric vector that contains the known side a, if known
b	numeric vector that contains the known side b, if known
c	numeric vector that contains the known side c (hypotenuse), if known

Details

Side `a` is the side adjacent to angle B and opposite angle A. Side `b` is the side adjacent to angle A and opposite angle B. Side c (hypotenuse) is opposite the right angle (angle C).

This function makes the following calculations:

1. the length of the missing side using the Pythagorean theorem,
2. the area of the right triangle,
3. the altitude of the right triangle,
4. the angle associated with the side named a (degrees),
5. the angle associated with the side named b (degrees), and
6. the angle associated with the side named c (degrees).

Value

`list` of known sides a, b, and c & the interior angles A, B, and C (right angle), in degrees, if and only if the given sides create a right triangle.

Author(s)

Irucka Embry

Source

1. r - Better error message for stopifnot? - Stack Overflow answered by Andrie on Dec 1 2011. See https://stackoverflow.com/questions/8343509/better-error-message-for-stopifnot.
2. r - switch() statement usage - Stack Overflow answered by Tommy on Oct 19 2011 and edited by Tommy on Mar 6 2012. See https://stackoverflow.com/questions/7825501/switch-statement-usage.
3. Using Switch Statement in R - Stack Overflow answered by Gavin Simpson on Jul 25 2013. See https://stackoverflow.com/questions/17847034/using-switch-statement-in-r.
4. r - How to not run an example using roxygen2? - Stack Overflow answered and edited by samkart on Jul 9 2017. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/12038160/how-to-not-run-an-example-using-roxygen2.
5. devtools - Issues in R package after CRAN asked to replace dontrun by donttest - Stack Overflow answered by Hong Ooi on Sep 1 2020. (Also see the additional comments in response to the answer.) See https://stackoverflow.com/questions/63693563/issues-in-r-package-after-cran-asked

References

1. r - Better error message for stopifnot? - Stack Overflow answered by Andrie on Dec 1 2011. See https://stackoverflow.com/questions/8343509/better-error-message-for-stopifnot.
2. Masoud Olia, Ph.D., P.E. and Contributing Authors, *Barron's FE (Fundamentals of Engineering Exam)*, 3rd Edition, Hauppauge, New York: Barron's Educational Series, Inc., 2015, page 44-45.
3. Wikimedia Foundation, Inc. Wikipedia, 28 December 2015, "Pythagorean theorem", https://en.wikipedia.org/wiki/Pythagorean_theorem.
4. Wikimedia Foundation, Inc. Wikipedia, 26 November 2015, "Radian", https://en.wikipedia.org/wiki/Radian.
5. Wikimedia Foundation, Inc. Wikipedia, 9 December 2015, "Right triangle", https://en.wikipedia.org/wiki/Right_triangle.

Examples

```
library(iemisc)

righttri(a = 3, b = 4, c = 5)




# See Source 4 and Source 5

library(iemisc)

try(righttri(0, 2)) # a = 0, b = 2

try(righttri(1, 2)) # a = 1, b = 2
```

```
try(righttri(a = 5, c = 10))

try(righttri(a = 3, c = 10))
```

---

rms   *Root-mean-square*

Description

This function computes the sample root-mean-square (RMS).

Usage

```
rms(x, na.rm = FALSE)
```

Arguments

x	numeric vector that contains the sample data points.
na.rm	logical vector that determines whether the missing values should be removed or not.

Details

RMS is expressed as

$$x_{rms} = \sqrt{\frac{\sum\limits_{i=1}^{n} x_i^2}{n}}$$

$x_r ms$ the sample harmonic mean

x the values in a sample

n the number of values

Value

sample root-mean-square as a numeric vector. The default choice is that any NA values will be kept (`na.rm = FALSE`). This can be changed by specifying `na.rm = TRUE`, such as `rms(x, na.rm = TRUE)`.

Author(s)

Irucka Embry

References

Masoud Olia, Ph.D., P.E. and Contributing Authors, *Barron's FE (Fundamentals of Engineering Exam)*, 3rd Edition, Hauppauge, New York: Barron's Educational Series, Inc., 2015, page 84.

See Also

sgm for geometric mean, shm for harmonic mean, cv for coefficient of variation (CV), relerror for relative error, approxerror for approximate error, and ranges for sample range.

Examples

```
library(iemisc)

samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34,
        65, 98, 3)

rms(samp)
```

---

sat_vapor_pressure *Saturation Vapor Pressure for Water*

Description

This function solves for the saturation vapor pressure of water using only the temperature of the water in either units of degrees Celsius, degrees Fahrenheit, or Kelvin.

Usage

```
sat_vapor_pressure(
  Temp,
  units = c("SI", "Eng", "Absolute"),
  formula = c("Huang", "Buck", "IAPWS")
)
```

Arguments

Temp	numeric vector that contains the temperature (degrees Celsius, degrees Fahrenheit, or Kelvin)
units	character vector that contains the system of units (options are SI for International System of Units, Eng for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), or Absolute for Absolute Units)

`formula` character vector that contains the source of the formula used to compute the saturation vapor pressure (options are Huang, Buck, IAPWS)

Details

The simplified equation is expressed as

$$P_s = \exp \frac{34.494 - \frac{4924.99}{t+237.1}}{(t + 105)^1.57}$$

for (t > 0 C)

P_s the saturation vapor pressure (Pa or psi)

t the water temperature, degrees Celsius

Value

the saturation vapor pressure for water as a numeric vector. The unit for `SI` and `Absolute` is Pascal (Pa), but the unit for `Eng` is pounds per square inch (psi). The units are not returned.

Note

Note: Please refer to the references for the formulas (Huang = Reference 1, IAPWS = Reference 2, and Buck = Reference 3)

Note: Please refer to the iemisc: Comparing Saturated Vapor Pressure Formulas to the Reference vignette for the comparisons to the reference saturated vapor pressure

Author(s)

Irucka Embry

References

1. Huang, J. (2018). "A Simple Accurate Formula for Calculating Saturation Vapor Pressure of Water and Ice", *Journal of Applied Meteorology and Climatology*, 57(6), 1265-1272. Retrieved Nov 4, 2021, https://web.archive.org/web/20221024040058/https://journals.ametsoc.org/view/journals/apme/57/6/jamc-d-17-0334.1.xml. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.
2. The International Association for the Properties of Water and Steam. IAPWS SR1-86(1992). "Revised Supplementary Release on Saturation Properties of Ordinary Water Substance", September 1992, http://www.iapws.org/relguide/Supp-sat.pdf
3. Holger Vömel, National Center for Atmospheric Research Earth Observing Laboratory, "Saturation vapor pressure formulations", https://web.archive.org/web/20170623040102/http://cires1.colorado.edu/~voemel/vp.html. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Example 1 - Example from the hydraulics package

library(iemisc)

vps <- hydraulics::svp(T = 10, units = "SI"); vps

vps2 <- sat_vapor_pressure(Temp = 10, units = "SI", formula = "Huang"); vps2




# Example 2 - from the Huang Reference

library(iemisc)

sat_vapor_pressure(Temp = c(0.01, seq(from = 20, to = 100, by = 20)), units
= "SI", formula = "Huang")




# Example 3 - compare with saturation_pressure_H2O from aiRthermo

install.load::load_package("iemisc", "units")

Temp <- 40

# create a numeric vector with the units of degrees Celsius
T_C <- set_units(Temp, "degree_C")
T_C

# create a numeric vector to convert from degrees Celsius to Kelvin
T_K <- T_C
T_K

# create a numeric vector with the units of Kelvin
units(T_K) <- make_units(K)

pre <- aiRthermo::saturation_pressure_H2O(drop_units(T_K))
pre

sat_vapor_pressure(Temp = drop_units(T_K), units = "Absolute", formula = "Huang")
```

`sat_vapor_pressure_ice` *Saturation Vapor Pressure for Ice*

Description

This function solves for the saturation vapor pressure of ice using only the temperature of the water in either units of degrees Celsius, degrees Fahrenheit, or Kelvin.

Usage

```
sat_vapor_pressure_ice(Temp, units = c("SI", "Eng", "Absolute"))
```

Arguments

`Temp`	numeric vector that contains the temperature (degrees Celsius, degrees Fahrenheit, or Kelvin)
`units`	character vector that contains the system of units [options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), or `Absolute` for Absolute Units]

Details

The simplified equation is expressed as

$$P_s = \exp \frac{43.494 - \frac{6545.8}{t+278}}{(t+868)^2}$$

for (t <= 0 C)

P_s the saturation vapor pressure (Pa or psi)

t the ice temperature, degrees Celsius

Value

the saturation vapor pressure for ice as a numeric vector. The unit for `SI` and `Absolute` is Pascal (Pa), but the unit for `Eng` is pounds per square inch (psi). The units are not returned.

Author(s)

Irucka Embry

References

Huang, J. (2018). "A Simple Accurate Formula for Calculating Saturation Vapor Pressure of Water and Ice", *Journal of Applied Meteorology and Climatology*, 57(6), 1265-1272. Retrieved Nov 4, 2021, https://web.archive.org/web/20221024040058/https://journals.ametsoc.org/view/journals/apme/57/6/jamc-d-17-0334.1.xml. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN. Used the Internet Archive: Wayback Machine archived version for acceptance into CRAN.

Examples

```
# Example from the Reference

library(iemisc)

sat_vapor_pressure_ice(Temp = seq(from = -100, to = 0, by = 20), units = "SI")
```

---

sec *Secant (in radians)*

Description

Calculates the value of secant for each element of x in radians.

Usage

```
sec(x)
```

Arguments

x	A numeric or complex vector containing values in radians

Value

The secant of each element of x in radians.

Note

Note: If you have a degree angle value, use secd instead.

Author(s)

Irucka Embry

Examples

```
library(iemisc)

# Examples

sec (seq(-2, 2, by = 1) * pi)

sec ((3 * pi) / 4)

sec (c((7/3) * pi, (5/2) * pi))
```

---

secd — *Secant (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of secant for each element of x in degrees in a manner compatible with GNU Octave/MATLAB.

Usage

```
secd(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The secant of each element of x in degrees.

Note

Note: If you have a radian (rad) angle value, use sec instead.

Author(s)

David Bateman (GNU Octave secd), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 553.

Examples

```
library(iemisc)

# Examples from GNU Octave secd
secd (seq(0, 80, by = 10))

secd (c(0, 180, 360))

secd (c(90, 270))
```

---

secprop *Section Properties Calculator [GNU Octave/MATLAB]*

Description

This function "is a short but really useful script that calculates the section properties for an arbitrary shape with holes." Reference: Caprani

Usage

```
secprop(
  outer_coord,
  inner_coord,
  original_plot = c(0, 1),
  final_plot = c(0, 1)
)
```

Arguments

outer_coord	numeric matrix that contains the "outer coordinates (x, y)" Reference: Caprani
inner_coord	numeric matrix that contains the "coordinates for a void" Reference: Caprani
original_plot	integer vector that contains 0, 1 only. 0 represents do not print the original plot and 1 is for printing the original plot. The default value is 0 (no original plot).
final_plot	integer vector that contains 0, 1 only. 0 represents do not print the final (transformed) plot and 1 is for printing the final (transformed) plot. The default value is 1 (final plot will be shown).

Value

SP numeric vector that contains "A (Area), I (Second Moment of Area), yt and yb properties of the section." Reference: Caprani

Note

Note: Please refer to the iiemisc: secprop Example (R and GNU Octave) vignette for the examples

Author(s)

Colin Caprani (secprop MATLAB function), Irucka Embry (secprop R function)

Source

loops - r - foreach unable to find object within function - Stack Overflow answered by sumshyftw on Jun 7 2019. See https://stackoverflow.com/questions/56498824/r-foreach-unable-to-find-object-within-fu 56499056.

References

Colin Caprani, "Section Properties Calculator", https://www.colincaprani.com/programming/matlab/.

`sgm` *Geometric mean*

Description

This function computes the sample geometric mean.

Usage

```
sgm(x, na.rm = FALSE)
```

Arguments

`x`	numeric vector that contains the sample data points (any negative values will be ignored).
`na.rm`	logical vector that determines whether the missing values should be removed or not.

Details

Geometric mean is expressed as

$$\bar{x}_g = (x_1 x_2 \cdots x_n)^{\frac{1}{n}}$$

$\bar{x}_g$ the sample geometric mean

$\boldsymbol{x}$ the values in a sample

$\boldsymbol{n}$ the number of positive values

"The geometric mean is used in averaging values that represent a rate of change. It is the positive nth root of the product of the n values."

Value

sample geometric mean as a numeric vector. The default choice is that any NA values will be kept (`na.rm = FALSE`). This can be changed by specifying `na.rm = TRUE`, such as `sgm(x, na.rm = TRUE)`.

Author(s)

Irucka Embry

References

Nathabandu T. Kottegoda and Renzo Rosso, *Statistics, Probability, and Reliability for Civil and Environmental Engineers*, New York City, New York: The McGraw-Hill Companies, Inc., 1997, page 13.

See Also

`mean` for arithmetic mean

`shm` for harmonic mean, `cv` for coefficient of variation (CV), `relerror` for relative error, `approxerror` for approximate error, `rms` for root-mean-square (RMS), and `ranges` for sample range.

Examples

```
# Example 1.13 from Kottegoda (page 13)

library(iemisc)

city_pop <- c(230000, 310000)
sgm(city_pop)

# Compare the geometric mean to the arithmetic mean
mean(city_pop)
```

`shm` *Harmonic mean*

Description

This function computes the sample harmonic mean.

Usage

```
shm(x, na.rm = FALSE)
```

Arguments

x	numeric vector that contains the sample data points.
na.rm	logical vector that determines whether the missing values should be removed or not.

Details

Harmonic mean is expressed as

$$\bar{x}_h = \frac{1}{\left(\frac{1}{n}\right)\left[\left(\frac{1}{x_1}\right) + \left(\frac{1}{x_2}\right) + \cdots + \left(\frac{1}{x_n}\right)\right]}$$

$\bar{x}_h$ the sample harmonic mean

x the values in a sample

n the number of values

"The harmonic mean is the reciprocal of the mean of the reciprocals. It is applied in situations where the reciprocal of a variable is averaged."

Value

sample harmonic mean as a numeric vector. The default choice is that any NA values will be kept (`na.rm = FALSE`). This can be changed by specifying `na.rm = TRUE`, such as `shm(x, na.rm = TRUE)`.

Author(s)

Irucka Embry

References

Nathabandu T. Kottegoda and Renzo Rosso, *Statistics, Probability, and Reliability for Civil and Environmental Engineers*, New York City, New York: The McGraw-Hill Companies, Inc., 1997, page 13.

See Also

`mean` for arithmetic mean

`sgm` for geometric mean, `cv` for coefficient of variation (CV), `relerror` for relative error, `approxerror` for approximate error, `rms` for root-mean-square (RMS), and `ranges` for sample range.

Examples

```
# Example 1.12 from Kottegoda (page 13)

install.load::load_package("iemisc", "data.table")

x <- c(0.20, 0.24, 0.16) # stream velocities in m/s
shm(x)
```

```
# using a matrix of the numeric vector x
mat1 <- matrix(data = x, nrow = length(x), ncol = 1, byrow = FALSE,
        dimnames = list(c(rep("", length(x))), "Velocities"))
shm(mat1)


# using a data.frame of the numeric vector x
df1 <- data.frame(x)
shm(df1)


# using a data.table of the numeric vector x
df2 <- data.table(x)
shm(df2)
```

`SimpIntCharg` *Simple Interest Charged (Engineering Economics)*

Description

Computes the total interest paid at the end of n periods using simple interest

Usage

```
SimpIntCharg(P, n, i)
```

Arguments

P	numeric vector that contains the present value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as whole number or decimal

Details

Simple Interest Charged is expressed as

$$I = Pni$$

P the "principal amount (lent or borrowed)"

I the "simple interest"

i the "interest rate per interest period"

n the "number of interest periods"

Value

SimpIntCharg numeric vector that contains the simple interest amount paid at the end of n periods rounded to 2 decimal places

Author(s)

Irucka Embry

References

1. Chinyere Onwubiko, *An Introduction to Engineering*, Mission, Kansas: Schroff Development Corporation, 1997, page 205-206.
2. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 116.

Examples

```
library(iemisc)

# Example for equation 4-1 from the Sullivan Reference text (page 116)
# Modified example to provide the simple interest amount paid only

SimpIntCharg(P = 1000, n = 3, i = 10) # the interest rate is 10%
```

SimpIntPaid *Simple Interest Paid (Engineering Economics)*

Description

Computes the total amount paid at the end of n periods using simple interest

Usage

```
SimpIntPaid(P, n, i)
```

Arguments

P	numeric vector that contains the present value(s)
n	numeric vector that contains the period value(s)
i	numeric vector that contains the interest rate(s) as whole number or decimal

Details

Simple Interest is expressed as

$$I = Pni$$

$$S_n = P + I$$

or

$$S_n = P\left(1 + ni\right)$$

P the "principal amount (lent or borrowed)"

S_n the "total amount paid back"

I the "simple interest"

i the "interest rate per interest period"

n the "number of interest periods"

Value

SimpIntPaid numeric vector that contains the total amount paid at the end of n periods rounded to 2 decimal places

Author(s)

Irucka Embry

References

1. Chinyere Onwubiko, *An Introduction to Engineering*, Mission, Kansas: Schroff Development Corporation, 1997, page 205-206.
2. William G. Sullivan, Elin M. Wicks, and C. Patrick Koelling, *Engineering Economy*, Fourteenth Edition, Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2009, page 116.

Examples

```
library(iemisc)

# Example for equation 4-1 from the Sullivan Reference text (page 116)
SimpIntPaid(1000, 3, 10) # the interest rate is 10%
```

sind — *Sine (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of sine for each element of x in degrees in a manner compatible with GNU Octave/MATLAB. Zero is returned for any "elements where x / 180 is an integer." Reference: Eaton.

Usage

```
sind(x)
```

Arguments

x	A numeric vector containing values in degrees

Value

The sine of each element of x in degrees. Zero for any "elements where x / 180 is an integer."

Note

Note: If you have a radian (rad) angle value, use sin instead.

Author(s)

David Bateman (GNU Octave sind), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Page 553.

Examples

```
library(iemisc)

# Examples from GNU Octave sind
sind(seq(10, 90, by = 10))

sind(c(0, 180, 360))

sind(c(90, 270))
```

size — *Size of R objects (GNU Octave/MATLAB compatible)*

Description

Provides the dimensions of R objects in a manner compatible with GNU Octave/MATLAB. This function is the same as `size`, except this `size` can find the size of character vectors too. Some documentation from `size`.

Usage

```
size(x, k)
```

Arguments

`x`	An R object (array, vector, or matrix)
`k`	integer specifying a particular dimension

Value

"Return the number of rows and columns of the object x as a numeric vector. If given a second argument, `size` will return the size of the corresponding dimension." Source: Eaton.

Author(s)

Hans Werner Borchers (pracma size), Irucka Embry

Source

pracma size function definition - R package pracma created and maintained by Hans Werner Borchers. See `size`.

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. https://docs.octave.org/octave.pdf. Pages 47-48.

See Also

`dim`, `size`

Examples

```
# Example from GNU Octave ndims function reference

size(matlab::ones(4, 1, 2, 1))
```

`splitcomma` *Split Comma*

Description

This function takes a character string in the form of Last Name (Surname), First Name or Second String, First String and transposes that string to First Name Last Name (Surname) or First String Second String while removing the comma.

Usage

```
splitcomma(string)
```

Arguments

`string`	character vector that contains a phrase separated by a comma or not. If not, there is no change.

Value

the character `vector` in the form of First Name Last Name

Author(s)

Irucka Embry

Source

1. regex - Split on first comma in string - Stack Overflow answered by flodel on Apr 25 2012. See https://stackoverflow.com/questions/10309122/split-on-first-comma-in-string.
2. regex - r regexp - replace title and suffix in any part of string with nothing in large file (> 2 million rows) - Stack Overflow answered by Molx on Apr 16 2015. See https://stackoverflow.com/questions/29680131/r-regexp-replace-title-and-suffix-in-any-part-of-string-w

Examples

```
# Example 1

install.load::load_package("iemisc", "data.table")

dtxx <- data.table(Names = c("Cooler, Wine", "Juice, Fruit", "Hard Water",
"Hot Bath", "Set, Data"))

dtxx[, Corrected_Names := splitcomma(dtxx$Names)]

dtxx



# Example 2

xtrax <- "FALSER, BRATTIE & SIMX, AGONY"

splitcomma(xtrax)
```

---

splitremove *Split Remove*

Description

This function removes characters from a string based on a character vector named remove. This function can be used to remove prefixes, suffixes, titles, etc. from a given character vector. The function splits the string by empty spaces, dots, commas, and parentheses first & then it removes the items that are in the remove vector.

Usage

```
splitremove(string, remove)
```

Arguments

`string`	character vector that contains the text to keep and to remove
`remove`	character vector that contains the characters to remove from the string

Value

the revised character `vector` with the contents of remove removed from the string

Author(s)

Irucka Embry

Source

regex - r regexp - replace title and suffix in any part of string with nothing in large file (> 2 million rows) - Stack Overflow answered by Molx on Apr 16 2015. See https://stackoverflow.com/questions/29680131/r-regexp-replace-title-and-suffix-in-any-part-of-string-with-nothing-in-large.

Examples

```
# Example

install.load::load_package("iemisc", "data.table")

# create the list of items to remove from the text
remove <- c("mister", "sir", "mr", "madam", "mrs", "miss", "ms", "iv",
"iii", "ii", "jr", "sr", "md", "phd", "mba", "pe", "mrcp", "and", "&", "prof",
"professor", "esquire", "esq", "dr", "doctor")

names <- data.table(Named = c("Alfredy 'Chipp' Kahner IV",
"Denis G. Barnekdt III", "JERUEG, RICHARDS Z. MR.", "EDWARDST, HOWARDD K. JR."))

# first use split comma
names[, Corrected_Named := splitcomma(names$Named)]

names

names[, Corrected_Named := splitremove(names$Corrected_Named, remove)]

names
```

---

sp_gravity *Specific Volume*

Description

This function solves for the specific volume of a substance using only the substance's density and the density of water.

Usage

```
sp_gravity(
  rho_w,
  rho_s,
  units = c("SI", "Eng"),
  Eng_units = c("slug/ft^3", "lbm/ft^3")
)
```

Arguments

rho_w	numeric vector that contains the density of water
rho_s	numeric vector that contains the density of the substance
units	character vector that contains the system of units [options are SI for International System of Units or Eng for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]
Eng_units	character vector that contains the unit for the density of water [options are slug/ft^3 or lbm/ft^3]

Details

The equation is expressed as

$$SG = \frac{rho_s}{gamma_w} = \frac{rho_s}{rho_w}$$

SG specific volume (dimensionless)

\gamma_w unit weight or specific weight of water (N/m^3 or lbf/ft^3)

\rho_s substance density (mass divided by volume) [kg/m^3, slug/ft^3", or lbm/ft^3]

\rho_w water density (mass divided by volume) [kg/m^3, slug/ft^3", or lbm/ft^3]

Value

the specific volume as a numeric vector

Author(s)

Irucka Embry

References

1. Material Properties, 27 March 2022, "Sand – Density – Heat Capacity – Thermal Conductivity", https://material-properties.org/sand-density-heat-capacity-thermal-conductivity/
2. WikiEngineer, 27 March 2022, "Water Properties & Definitions", https://web.archive.org/web/20210412034245/http://www.wikiengineer.com/Water-Resources/PropertiesAndDefinitions. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Examples

install.load::load_package("iemisc", "units")

# The density of sand is 1500 kg/m^3 -- Reference 1

rho_sand <- set_units(1500, "kg/m^3")
```

```
# convert this density to slug/ft^3
rho_sand_slug <- rho_sand

# create a numeric vector with the units of slug/ft^3
units(rho_sand_slug) <- make_units(slug/ft^3)


# convert this density to lbm/ft^3
rho_sand_lbm <- rho_sand

# create a numeric vector with the units of lb/ft^3
units(rho_sand_lbm) <- make_units(lb/ft^3)



rho1 <- density_water(Temp = 68, units = "Eng", Eng_units = "slug/ft^3")

sp_gravity(rho_w = rho1, rho_s = rho_sand_slug, units = "Eng", Eng_units = "slug/ft^3")


rho2 <- density_water(Temp = 68, units = "Eng", Eng_units = "lbm/ft^3")

sp_gravity(rho_w = rho2, rho_s = rho_sand_lbm, units = "Eng", Eng_units = "lbm/ft^3")


rho3 <- density_water(Temp = 20, units = "SI")

sp_gravity(rho_w = rho3, rho_s = rho_sand, units = "SI")
```

---

sp_volume *Specific Volume*

Description

This function solves for the specific volume of a substance using only the substance's density

Usage

```
sp_volume(rho)
```

Arguments

rho	numeric vector that contains the density

Details

The equation is expressed as

$$nu = \frac{1}{rho}$$

\rho substance density (mass divided by volume) [kg/m^3, slug/ft^3", or lbm/ft^3]

Value

the specific volume as a numeric vector

Author(s)

Irucka Embry

References

WikiEngineer, 27 March 2022, "Water Properties & Definitions", https://web.archive.org/web/20210412034245/http://www.wikiengineer.com/Water-Resources/PropertiesAndDefinitions. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Examples

library(iemisc)

rho1 <- density_water(Temp = 68, units = "Eng", Eng_units = "slug/ft^3")

sp_volume(rho = rho1) # slug/ft^3


rho2 <- density_water(Temp = 68, units = "Eng", Eng_units = "lbm/ft^3")

sp_volume(rho = rho2) # lbm/ft^3


rho3 <- density_water(Temp = 20, units = "SI")

sp_volume(rho = rho3) # kg / m^3
```

`surface_area` *Calculate the Total Surface Area of Linear Surfaces*

Description

This function computes the total surface area of linear surfaces (total sum of width x length). This function was created to use in drainage area calculations; however, it can be used in other calculations as well.

Usage

```
surface_area(
  length = NULL,
  width = NULL,
  surface_area_table = NULL,
  lw_units = c("inch", "feet", "survey_foot", "yard", "mile", "centimeter", "meter",
    "kilometer")
)
```

Arguments

`length`	numeric vector containing the length value(s) in one of the lw_units values.
`width`	numeric vector containing the width value(s) in one of the lw_units values.
`surface_area_table`	data.frame/data.table/tibble, list, or matrix containing the length in column 1 and the width in column 2
`lw_units`	character vector containing the units for the length and the width (default = "feet"). The other possible units are "inch", "survey_foot", "yard", "mile", "centimeter", "meter", or "kilometer". The units should be consistent and not mixed.

Value

surface area as a numeric vector in the provided square units. The calculated value will be the same for all units in this function. The units specified in this function are used in the `rain_garden_driveway` function.

Author(s)

Irucka Embry

See Also

`rain_garden_driveway` for calculating the rain garden size for driveways

Examples

```
# Note: the units must be consistent

# Example 1

library(iemisc)

length1 <- c(220, 150, 30)
width1 <- c(75, 89, 80)
surface_area(width = width1, length = length1, lw_units = "meter")


# Example 2

library(iemisc)

length2 <- c(333, 681, 73)
width2 <- c(17.4, 9.5, 8)
surface_area_table = list(Length = length2, Width = width2)
surface_area(surface_area_table = surface_area_table, lw_units = "mile")
```

`surf_tens_water` *Water Surface Tension for Liquid Water*

Description

This function solves for the surface tension of water using only the temperature of the water in either units of degrees Celsius, degrees Fahrenheit, or Kelvin.

Usage

```
surf_tens_water(Temp, units = c("SI", "Eng", "Absolute"))
```

Arguments

`Temp`	numeric vector that contains the temperature (degrees Celsius, degrees Fahrenheit, or Kelvin)
`units`	character vector that contains the system of units (options are `SI` for International System of Units, `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom), or `Absolute` for Absolute Units)

Details

The simplified equation is expressed as

$$sigma = \frac{1}{a + bT + cT^2 + dT^3}$$

with

$$a = 0.075652711$$
$$b = -0.00013936956$$
$$c = -3.0842103 * 10^-7$$
$$d = 2.7588435 * 10^-10$$

\sigma Water Surface Tension (N/m or lbf/ft)

T the water temperature, degrees Celsius

Value

the surface tension as a numeric vector. The units are not returned.

Author(s)

Irucka Embry

References

C. O. Popiel & J. Wojtkowiak (1998). "Simple Formulas for Thermophysical Properties of Liquid Water for Heat Transfer Calculations (from 0C to 150C)". *Heat Transfer Engineering*, 19:3, 87-101, article from ResearchGate: https://www.researchgate.net/publication/239243539_Simple_Formulas_for_Thermophysical_Properties_of_Liquid_Water_for_Heat_Transfer_Calculations_from_0C_to_150C.

Examples

```
# Example (Compare to the tabulated values in the Reference paper)

install.load::load_package("iemisc", "data.table", "round")

Temp <- c(0, 0.01, 3.86, seq(5, 95, by = 5), 99.974, seq(100, 150, by = 5))

surface_tension <- data.table("Temperature (degrees C)" = Temp, "omega (N / m)"
= round_r3(surf_tens_water(Temp, units = "SI"), d = 5)); surface_tension
```

tand *Tangent (in degrees) [GNU Octave/MATLAB compatible]*

Description

Calculates the value of tangent for each element of `x` in degrees in a manner compatible with GNU Octave/MATLAB. Zero is returned for any "elements where `x` / 180 is an integer and `Inf` for elements where (`x` - 90) / 180 is an integer." Reference: Eaton.

Usage

```
tand(x)
```

Arguments

`x`	A numeric vector containing values in degrees

Value

The tangent of each element of `x` in degrees. Zero for any "elements where `x` / 180 is an integer and `Inf` for elements where (`x` - 90) / 180 is an integer."

Note

Note: If you have a radian (rad) angle value, use `tan` instead.

Author(s)

David Bateman (GNU Octave tand), Irucka Embry

References

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring (November 2022). *GNU Octave: A high-level interactive language for numerical computations: Edition 7 for Octave version 7.3.0*. `https://docs.octave.org/octave.pdf`. Page 553.

Examples

```
library(iemisc)

# Examples from GNU Octave tand
tand(seq(10, 80, by = 10))

tand(c(0, 180, 360))

tand(c(90, 270))
```

`uc_composite_CN` *Composite CN (Curve Number) with Unconnected Impervious Area*

Description

This function computes the composite CN (Curve Number) for unconnected impervious areas and total impervious areas less than 30 percent.

Usage

```
uc_composite_CN(pervious_CN, impervious, R)
```

Arguments

`pervious_CN`	numeric vector containing the pervious runoff curve number
`impervious`	numeric vector containing the percent imperviousness
`R`	numeric vector containing the ratio of unconnected impervious area to total impervious area

Value

the Composite Runoff Curve Number as a single numeric vector, in the range [0, 100]

Author(s)

Irucka Embry

References

United States Department of Agriculture Natural Resources Conservation Service Conservation Engineering Division, "Urban Hydrology for Small Watersheds Technical Release 55 (TR-55)", June 1986, pages 2-11 - 2-16, https://web.archive.org/web/20230810204711/https://directives.sc.egov.usda.gov/OpenNonWebContent.aspx?content=22162.wba [Recovered with the Internet Archive: Wayback Machine]

Examples

```
# Please refer to iemiscdata: Weighted CN Calculations Using the Composite CN
# vignette in the iemiscdata package
```

unit_wt — *Unit Weight or Specific Weight*

Description

This function solves for the unit weight or specific weight of a substance using only the substance's density and gravitational acceleration.

Usage

```
unit_wt(rho, units = c("SI", "Eng"), Eng_units = c("slug/ft^3", "lbm/ft^3"))
```

Arguments

`rho`	numeric vector that contains the density
`units`	character vector that contains the system of units [options are `SI` for International System of Units or `Eng` for English units (United States Customary System in the United States and Imperial Units in the United Kingdom)]
`Eng_units`	character vector that contains the density English units [options are `lbm/ft^3` or `slug/ft^3`]

Details

The equation is expressed as

$$gamma = \frac{m * g}{V} = rho * g = rho * \frac{g}{gc}$$

\gamma unit weight or specific weight (N/m^3 or lbf/ft^3)

m substance mass (kg, lbm, or slugs)

g gravitational acceleration (m/s^2 or ft/sec^2)

gc "unit conversion factor used to convert mass to force or vice versa" (lbm-ft/lbf-sec^2)

V the volume of the substance (m^3 or ft^3)

\rho substance density (mass divided by volume) [kg/m^3, slug/ft^3", or lbm/ft^3]

Value

the unit weight or specific weight as a numeric vector. The units are not returned.

Author(s)

Irucka Embry

References

1. The Engineering ToolBox, 27 March 2022, "Water - Density, Specific Weight and Thermal Expansion Coefficients", https://www.engineeringtoolbox.com/water-density-specific-weight-d_595.html
2. WikiEngineer, 27 March 2022, "Water Properties & Definitions", https://web.archive.org/web/20210412034245/http://www.wikiengineer.com/Water-Resources/PropertiesAndDefinitions. Retrieved thanks to the Internet Archive: Wayback Machine
3. Wikimedia Foundation, Inc. Wikipedia, 27 March 2022, "gc (engineering)", https://en.wikipedia.org/wiki/Gc_(engineering).
4. Wikimedia Foundation, Inc. Wikipedia, 27 March 2022, "Specific weight", https://en.wikipedia.org/wiki/Specific_weight.

Examples

```
# Examples

library(iemisc)

rho1 <- density_water(Temp = 68, units = "Eng", Eng_units = "slug/ft^3")

unit_wt(rho = rho1, units = "Eng", Eng_units = "slug/ft^3")


rho2 <- density_water(Temp = 68, units = "Eng", Eng_units = "lbm/ft^3")

unit_wt(rho = rho2, units = "Eng", Eng_units = "lbm/ft^3")


rho3 <- density_water(Temp = 20, units = "SI")

unit_wt(rho = rho3, units = "SI")
```

`volsphere` *Sphere volume*

Description

This function computes the volume of a sphere using a given radius.

Usage

```
volsphere(r)
```

Arguments

r numeric vector, matrix, data.frame, or data.table that contains the radius of a sphere.

Details

The radius of a sphere is "the integral of the surface area of a sphere."

Volume of a sphere is expressed as

$$V = \frac{4}{3}\pi r^3$$

V the volume of a sphere

r the radius of a sphere

Value

volume of a sphere (as L^3 units) as an R object: a numeric `vector` or a named numeric vector if using a named object (`matrix`, `data.frame`, or `data.table`).

Author(s)

Irucka Embry

References

Wikimedia Foundation, Inc. Wikipedia, 30 December 2015, "Volume", https://en.wikipedia.org/wiki/Volume.

Examples

```
install.load::load_package("iemisc", "data.table")

volsphere(3) # in

volsphere(4.5) # in


x <- c(3, 4, 0.2, 12, 34, 7.5) # cm
volsphere(x)


# using a matrix of the numeric vector x
mat1 <- matrix(data = x, nrow = length(x), ncol = 1, byrow = FALSE,
       dimnames = list(c(rep("", length(x))), "Radius"))
volsphere(mat1)


# using a data.frame of the numeric vector x
df1 <- data.frame(x)
```

```
volsphere(df1)


# using a data.table of the numeric vector x
df2 <- data.table(x)
volsphere(df2)
```

---

weighted_C *Calculate the Weighted C factor*

Description

This function computes the weighted C factor using the user-supplied unit or the default unit of an acre for use in the Rational Formula.

Usage

```
weighted_C(
  C = NULL,
  area = NULL,
  area_pct = NULL,
  area_units = c("acre", "square feet", "square mile", "hectare", "square kilometer"),
  C_area_table = NULL,
  C_area_pct_table = NULL
)
```

Arguments

`C`	numeric vector containing dimensionless C factor(s)
`area`	numeric vector containing the surface land area
`area_pct`	numeric vector containing the surface land area, as a percent (decimal or whole number)
`area_units`	character vector containing the units for area (default = "acre"). The other possible units are "square feet", "square mile", "hectare", or "square kilometer". The units should be consistent and not mixed.
`C_area_table`	data.frame/data.table/tibble, list, or matrix containing the C in column 1 and the area in column 2
`C_area_pct_table`	data.frame/data.table/tibble, list, or matrix containing the C in column 1 and the area_pct in column 2

Value

Weighted C factor as a single numeric vector, in the range [0, 1]

Author(s)

Irucka Embry

References

Engineering Hydrology Training Series Module 104 - Runoff Curve Number Computations Study Guide, United States Department of Agriculture Soil Conservation Service National Employee Development Staff, September 1989, page 21 https://web.archive.org/web/20210414043852/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/training/runoff-curve-numbers1.pdf. Retrieved thanks to the Internet Archive: Wayback Machine

Examples

```
# Note: the default area unit is acre

# Example 1

library(iemisc)

area1 <- c(220, 150, 30)
C1 <- c(75, 89, 80)
weighted_C(C = C1, area = area1)


# Example 2

library(iemisc)

area2 <- c(220, 150, 30)
area_pct2 <- area2 / sum(area2)
C2 <- c(80, 95, 80)
C_area_pct_table2 <- data.frame(C2, area_pct2)
weighted_C(C_area_pct_table = C_area_pct_table2)


# Example 3

install.load::load_package("iemisc", "data.table")

C_area_table3 <- data.table(C = c(98, 100, 45), area = c(2.53, 453.00, 0.21))
weighted_C(C_area_table = C_area_table3)


# Example 4

library(iemisc)

C4 <- c(98, 100, 45)
area_pct4 <- c(0.15, 0.23, 0.62)
weighted_C(C = C4, area_pct = area_pct4)
```

```
# Example 5

library(iemisc)

data_matrix5a <- matrix(c(98, 30, 40, 43, 57, 3.24, 1, 30, 50, 123), nrow = 5,
ncol = 2, dimnames = list(rep("", 5), c("C", "Area")))
weighted_C(C_area_table = data_matrix5a)


# using ramify to create the matrix

import::from(ramify, mat)

data_matrix5b <- mat("98 30 40 43 57;3.24 1 30 50 123", rows = FALSE, sep = " ",
dimnames = list(rep("", 5), c("C", "Area")))
weighted_C(C_area_table = data_matrix5b)


# Example 6 - using area in square feet

library(iemisc)

data_list6 <- list(C = c(77, 29, 68), Area = c(43560, 56893, 345329.32))
weighted_C(C_area_table = data_list6, area_units = "square feet")


# Example 7

install.load::load_package("iemisc", "data.table")

# Impervious area - 3.04 acre
# 45% of total area
# 0.80 C Factor

# Pervious area - 4.67 acre
# 55% of total area
# 0.20 C factor

C_area_table7 <- data.table(C = c(0.80, 0.20), area = c(3.04, 4.67))
weighted_C(C_area_table = C_area_table7)


# Example 8

# Impervious area - 2.44 acre
# 32% of total area
# 0.80 C Factor

# Pervious area - 5.03 acre
# 68% of total area
# 0.20 C factor

C8 <- c(0.80, 0.20)
```

```
area_pct8 <- c(0.32, 0.68)
weighted_C(C = C8, area_pct = area_pct8)


# Example 9

library(iemisc)

# Medium density residential area - 30 hectares (75.0% of total area),
# 0.31 - 0.40 C factor
# High density residential area - 3 hectares (7.50% of total area),
# 0.49 - 0.60 C factor
# Agricultural area - 7 hectares (17.5% of total area), 0.15 - 0.21 C factor

C3 <- c(mean(seq(0.31, 0.40, by = 0.01)), mean(seq(0.49, 0.60, by = 0.01)),
mean(seq(0.15, 0.21, by = 0.01)))
area3 <- c(30, 3, 7)
weighted_C(C = C3, area = area3, area_units = "hectare")
```

---

weighted_CN *Calculate the Weighted CN (Curve Number)*

Description

This function computes the weighted CN (Curve Number) using the user-supplied unit or the default unit of an acre.

Usage

```
weighted_CN(
  CN = NULL,
  CN_area_table = NULL,
  CN_area_pct_table = NULL,
  area = NULL,
  area_pct = NULL,
  area_units = c("acre", "square feet", "square mile", "hectare", "square kilometer")
)
```

Arguments

CN	numeric vector containing dimensionless Curve Number(s)
CN_area_table	data.frame/data.table/tibble, list, or matrix containing the CN in column 1 and the area in column 2

CN_area_pct_table
: data.frame/data.table/tibble, list, or matrix containing the CN in column 1 and the area_pct in column 2

area
: numeric vector containing the surface land area

area_pct
: numeric vector containing the surface land area, as a percent (decimal or whole number)

area_units
: character vector containing the units for area (default = "acre"). The other possible units are "square feet", "square mile", "hectare", or "square kilometer". The units should be consistent and not mixed.

Value

the Weighted Curve Number as a single numeric vector, in the range [0, 100]

Note

This function was originally part of Claudia Vitolo's curvenumber package that Irucka now maintains.

Author(s)

Irucka Embry

References

1. United States Department of Agriculture Soil Conservation Service National Employee Development Staff, "Engineering Hydrology Training Series Module 104 - Runoff Curve Number Computations Study Guide", September 1989, page 21, https://web.archive.org/web/20210414043852/https://www.wcc.nrcs.usda.gov/ftpref/wntsc/H&H/training/runoff-curve-numbers1.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
2. Dr. Clyde Munster, P.E., Texas A&M University Department of Biological and Agricultural Engineering, "Rational Method: Calculating Peak Flow Rates", https://web.archive.org/web/20180218221234/http://munster.tamu.edu/Study_Abroad/BAEN460_AGSM335/PowerPoint/RationalMethod_5.pdf. Retrieved thanks to the Internet Archive: Wayback Machine
3. United States Department of Agriculture Natural Resources Conservation Service Conservation Engineering Division, "Urban Hydrology for Small Watersheds Technical Release 55 (TR-55)", June 1986, https://web.archive.org/web/20230810204711/https://directives.sc.egov.usda.gov/OpenNonWebContent.aspx?content=22162.wba [Recovered with the Internet Archive: Wayback Machine]

Examples

```
# Note: the default area unit is an acre

# Example 1

library(iemisc)

area1 <- c(220, 150, 30)
```

```
CN1 <- c(75, 89, 80)
weighted_CN(CN = CN1, area = area1)


# Example 2

library(iemisc)

area2 <- c(220, 150, 30)
area_pct2 <- area2 / sum(area2)
CN2 <- c(80, 95, 80)
CN_area_pct_table2 <- data.frame(CN2, area_pct2)
weighted_CN(CN_area_pct_table = CN_area_pct_table2)


# Example 3

install.load::load_package("iemisc", "data.table")

CN_area_table3 <- data.table(CN = c(98, 100, 45), area = c(2.53, 453.00, 0.21))
weighted_CN(CN_area_table = CN_area_table3)


# Example 4

library(iemisc)

CN4 <- c(98, 100, 45)
area_pct4 <- c(0.15, 0.23, 0.62)
weighted_CN(CN = CN4, area_pct = area_pct4)


# Example 5

library(iemisc)

import::from(ramify, mat)


data_matrix5a <- matrix(c(98, 30, 40, 43, 57, 3.24, 1, 30, 50, 123),
nrow = 5, ncol = 2, dimnames = list(rep("", 5), c("C", "Area")))
weighted_CN(CN_area_table = data_matrix5a)


# using ramify to create the matrix
data_matrix5b <- mat("98 30 40 43 57;3.24 1 30 50 123", rows = FALSE,
sep = " ", dimnames = list(rep("", 5), c("CN", "Area")))
weighted_CN(CN_area_table = data_matrix5b)


# Example 6 - using area in square feet

library(iemisc)
```

```
data_list6 <- list(CN = c(77, 29, 68), Area = c(43560, 56893, 345329.32))
weighted_CN(CN_area_table = data_list6, area_units = "square feet")


# Example 7 - using area in whole percents

library(iemisc)

CN7 <- c(61, 74)
area_pct7 <- c(30, 70)
weighted_CN(CN = CN7, area_pct = area_pct7)
```

`%//%` *Floor Division (Python compatible)*

Description

Performs floor (or integer) division on 2 numeric vectors.

Usage

```
x %//% y
```

Arguments

x	numeric vector
y	numeric vector

Value

The integral part of the quotient is returned.

Author(s)

Irucka Embry

References

Jakub Przywóski: "Python Reference (The Right Way)". // floor division, 2015, Revision 9a3b94e7, https://python-reference.readthedocs.io/en/latest/docs/operators/floor_division.html.

Examples

```
# Example 1 -- From the Python reference

library(iemisc)

5.0 / 2
# 2.5

5.0 %//% 2
# 2.0
```

`%inorder%` *IN ORDER for Character and Numeric Vectors*

Description

This function preserves the original order (sequence) of character vectors.

Usage

```
y %inorder% table
```

Arguments

`y`	numeric vector that contains the sequence to return
`table`	character vector, data.table, and/or tibble that has the character values to be checked within

Value

character vector with the characters in the original sequence

Author(s)

John Wallace (Stack Overflow R code), Irucka Embry

Source

R - preserve order when using matching operators (

Examples

```
# Examples (from the Source)

LETTERS[1:26 %in% 4:1]

LETTERS[1:26 %inorder% 4:1]



LETTERS[1:26 %in% 3:-5]

LETTERS[1:26 %inorder% 3:-5]


data.frame(letters, LETTERS)[1:5 %in% 3:-5, ]

data.frame(letters, LETTERS)[1:5 %inorder% 3:-5, ]


library(data.table)

data.table(letters, LETTERS)[1:5 %inorder% 3:-5, ]


library(tibble)

tibble(letters, LETTERS)[1:5 %inorder% 3:-5, ]
```

`%notchin%` *Not CHIN or IN for Character and Numeric Vectors*

Description

This function performs a quick, case sensitive search of character vectors that are not in a set of character vectors or a quick, search of numeric vectors that are not in a set of numeric vectors using `Negate` chin for character vectors and `Negate in` for numeric vectors

Usage

```
x %notchin% y
```

Arguments

x	character or numeric vector that contains the values to not be matched
y	character or numeric vector that has the values to be checked within

Details

'Utilizes chin from data.table to quickly complete a case insensitive search through a character vector to return a logical vector of string detections. Will always return TRUE or FALSE for each position of string regardless of NA missing values in either provided vector. NA in string will never match an NA value in the vector.'

Value

logical vector the length of the original string (x). TRUE means that x is not in y and FALSE means that x is in y

Author(s)

kaijagahm (R code), Irucka Embry

Source

The

Examples

```
# Examples

x <- c("apple", "banana", "cherry", NA)

"apple" %notchin% x

c("apple", "BANANA", "coconut", NA) %notchin% x

x


"a" %notchin% letters[5:20]
letters[5:20] %notchin% "a"


"a" %notchin% LETTERS
LETTERS %notchin% "a"


1 %notchin% -12:20
-12:20 %notchin% 1
```

%qsin% *Quick Search*

Description

This function performs a quick, case insensitive search of strings

Usage

```
string %qsin% vector
```

Arguments

`string`	character vector that contains the values to match
`vector`	character vector that has the values to be matched against

Details

'Utilizes chin from data.table to quickly complete a case insensitive search through a character vector to return a logical vector of string detections. Will always return TRUE or FALSE for each position of string regardless of NA missing values in either provided vector. NA in string will never match an NA value in the vector.'

Value

logical vector the length of the original string

Author(s)

Dylan Russell (stackoverflow code, examples, and function details, function description), Irucka Embry

Source

data.table - R case-insensitive %in% - Stack Overflow answered and edited by Dylan Russell on Sep 13, 2020. See https://stackoverflow.com/questions/63874824/r-case-insensitive-in.

Examples

```
# Examples

x <- c("apple", "banana", "cherry", NA)

"apple" %qsin% x

c("APPLE", "BANANA", "coconut", NA) %qsin% x
```

Index