Package 'ConSciR'

September 10, 2025

Type Package

Title Tools for Conservation Science

Version 0.2.0

Maintainer Bhavesh Shah <bhaveshshah01@gmail.com>

Description Provides data science tools for conservation science, including methods for environmental analysis applications, humidity calculations, sustainability metrics, engineering calculations, and data visualisation. Supports conservators, scientists, and engineers working with cultural heritage data.
The package was motivated by the developing tools and frameworks outlined in Cosaert and Beltran et al. (2022) ``Tools for the Analysis of Collection Environments" <https://www.getty.edu/conservation/publications_resources/pdf_publications/tools_for_the_analysis_of_collection_environments.html>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

VignetteBuilder knitr

URL https://bhavshah01.github.io/ConSciR/, https://github.com/BhavShah01/ConSciR

BugReports https://github.com/BhavShah01/ConSciR/issues

Config/testthat/edition 3

Suggests knitr, rmarkdown, testthat (>= 3.0.0), bslib

Imports stats, tools, tidyr, readr, readxl, stringr, rlang, shiny, ggplot2, dplyr, lubridate, padr

Depends R (>= 4.1.0)

NeedsCompilation no

Author Bhavesh Shah [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0001-8673-0589>),
Annelies Cosaert [aut] (ORCID: <https://orcid.org/0009-0004-1269-4465>),
Vincent Beltran [aut],
Emily R Long [ctb] (ORCID: <https://orcid.org/0000-0003-3162-4521>),
Marcin Zygmunt [ctb] (ORCID: <https://orcid.org/0000-0003-1304-8591>)

Repository CRAN

Date/Publication 2025-09-10 08:50:24 UTC

Contents

`calcAD` *Calculate Air Density*

Description

Function to calculate air density based on temperature (°C), relative humidity in (%), and atmospheric pressure (hPa).

Usage

```
calcAD(Temp, RH, P_atm = 1013.25, R_dry = 287.058, R_vap = 461.495, ...)
```

Arguments

`Temp`	Temperature (°Celsius)
`RH`	Relative Humidity (0-100%)
`P_atm`	Atmospheric pressure = 1013.25 (hPa)
`R_dry`	Specific gas constant for dry air = 287.058 (J/(kg·K))
`R_vap`	Specific gas constant for water vapor = 461.495 (J/(kg·K))
`...`	Addtional arguments to supply to `calcPws`

Value

Air density in kg/m³

See Also

`calcMR` for calculating mixing ratio

`calcAD` for calculating air density

`calcPw` for calculating water vapour pressure

`calcPws` for calculating water vapour saturation pressure

Examples

```
calcAD(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(AirDensity = calcAD(Temp, RH))
```

`calcAH` *Calculate Absolute Humidity*

Description

Function to calculate the absolute humidity (g/m³) from temperature (°C) and relative humidity (%).

Absolute humidity is the mass of water in a unit volume of air at a given temperature and pressure.

Usage

```
calcAH(Temp, RH, P_atm = 1013.25)
```

Arguments

`Temp`	Temperature (°Celsius)
`RH`	Relative Humidity (0-100%)
`P_atm`	Atmospheric pressure = 1013.25 (hPa)

Value

AH Absolute Humidity (g/m³)

References

Buck, A. L. (1981). New equations for computing vapor pressure and enhancement factor. Journal of Applied Meteorology, 20(12), 1527-1532.

Examples

```
calcAH(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)
```

```
mydata |> dplyr::mutate(Abs = calcAH(Temp, RH))
```

`calcCoolingCapacity` *Calculate Cooling Capacity*

Description

This function calculates the required cooling capacity based on power consumption, power factor, safety factor, and efficiency.

Cooling capacity is the amount of energy transferred during a cooling process.

Usage

```
calcCoolingCapacity(
  Power,
  power_factor = 0.85,
  safety_factor = 1.2,
  efficiency = 0.7
)
```

Arguments

`Power`	Power consumption in Watts (W).
`power_factor`	Power factor, default is 0.85.
`safety_factor`	Safety factor, default is 1.2 (20% extra capacity).
`efficiency`	Efficiency of the cooling system, default is 0.7 (70%).

Value

Required cooling capacity in kilowatts (kW).

Examples

```
calcCoolingCapacity(1000)

calcCoolingCapacity(1500, power_factor = 0.9, safety_factor = 1.3, efficiency = 0.8)
```

`calcCoolingPower` *Calculate Cooling Power*

Description

This function calculates the cooling power based on initial and final air conditions and volume flow rate.

Cooling power is the rate of energy transferred during a cooling process.

Usage

```
calcCoolingPower(Temp1, Temp2, RH1, RH2, volumeFlowRate)
```

Arguments

`Temp1`	Initial Temperature (°Celsius)
`Temp2`	Final Temperature (°Celsius)
`RH1`	Initial Relative Humidity (0-100%)
`RH2`	Final Relative Humidity (0-100%)
`volumeFlowRate`	Volume flow rate of air (m^3/s)

Value

Cooling power in kilowatts (kW)

References

ASHRAE Handbook Fundamentals

See Also

`calcEnthalpy`, `calcAD`

Examples

```
calcCoolingPower(30, 22, 70, 55, 0.8)

calcCoolingPower(Temp1 = 25, Temp2 = 20, RH1 = 70, RH2 = 50, volumeFlowRate = 0.5)
```

`calcDP` *Calculate Dew Point*

Description

Function to calculate dew point (°C) from temperature (°C) and relative humidity (%).

The dew point is the temperature at which air becomes saturated with moisture and water vapour begins to condense.

Usage

```
calcDP(Temp, RH, method = c("Magnus", "Buck"))
```

Arguments

`Temp`	Temperature (°Celsius)
`RH`	Relative Humidity (0-100%)
`method`	Character; formula to use, either `"Magnus"` or `"Buck"`. Defaults to `"Magnus"`.

Details

This function supports two methods for dew point calculation:

- `"Magnus"` (default): Uses the August-Roche-Magnus approximation, valid for 0°C < Temp < 60°C and 1% < RH < 100%.
- `"Buck"`: Uses the Arden Buck equation with Bögel modification, valid for -30°C < Temp < 60°C and 1% < RH < 100%.

Both methods compute saturation vapour pressure and convert relative humidity to dew point temperature. The Magnus method is chosen as the default because it is more stable when used with the `calcTemp` and `calcRH_DP` functions.

Value

Td (DP), Dew Point (°Celsius)

Note

More details of the equations are also available in the source R code.

References

Alduchov, O. A., and R. E. Eskridge, 1996: Improved Magnus' form approximation of saturation vapor pressure. J. Appl. Meteor., 35, 601–609

Buck, A. L., 1981: New Equations for Computing Vapor Pressure and Enhancement Factor. J. Appl. Meteor. Climatol., 20, 1527–1532, https://doi.org/10.1175/1520-0450(1981)020<1527:NEFCVP>2.0.CO;2.

Buck (1996), Buck (1996), Buck Research CR-1A User's Manual, Appendix 1.

https://bmcnoldy.earth.miami.edu/Humidity.html

See Also

calcTemp for calculating temperature

calcRH_DP for calculating relative humidity from dew point

calcDP for calculating dew point

calcRH_AH for calculating relative humidity from absolute humidity

Examples

```
# Default Magnus method
calcDP(20, 50)

# Using Buck method
calcDP(20, 50, method = "Buck")

# Validation check
calcDP(20, calcRH_DP(20, calcDP(20, 50)))
calcDP(20, calcRH_DP(20, calcDP(20, 50, method = "Buck"), method = "Buck"), method = "Buck")


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |>
  dplyr::mutate(
    DewPoint = calcDP(Temp, RH),
    DewPoint_Buck = calcDP(Temp, RH, method = "Buck"))
```

---

calcEMC_wood *Calculate Equilibrium Moisture Content for wood (EMC)*

Description

This function calculates the Equilibrium Moisture Content (EMC) of wood based on relative humidity and temperature.

Equilibrium Moisture Content (EMC) is the moisture content at which a material, such as wood or other hygroscopic substanceshas reached an equilibrium with its environment and is no longer gaining or losing moisture under specific temperature and relative humidity.

Usage

```
calcEMC_wood(Temp, RH)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)

Value

EMC, Equilibrium Moisture Content (0-100%)

References

Simpson, W. T. (1998). Equilibrium moisture content of wood in outdoor locations in the United States and worldwide. Res. Note FPL-RN-0268. Madison, WI: U.S. Department of Agriculture, Forest Service, Forest Products Laboratory.

Hailwood, A. J., and Horrobin, S. (1946). Absorption of water by polymers: Analysis in terms of a simple model. Transactions of the Faraday Society 42, B084-B092. DOI:10.1039/TF946420B084

Examples

```
calcEMC_wood(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(EMC = calcEMC_wood(Temp, RH))
```

---

calcEnthalpy *Calculate Enthalpy*

Description

Function to calculate enthalpy from temperature (°C) and relative humidity (%).

Enthalpy is the total heat content of air, combining sensible (related to temperature) and latent heat (related to moisture content), used in HVAC calculations. Enthalpy is the amount of energy required to bring a gas to its current state from a dry gas at 0°C.

Usage

```
calcEnthalpy(Temp, RH, ...)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)
...	Additional arguments to supply to calcPws and calcMR

Value

h Enthalpy (kJ/kg)

Examples

```
calcEnthalpy(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(Enthalpy = calcEnthalpy(Temp, RH))
```

---

calcFP *Calculate Frost Point*

Description

Function to calculate frost point (°C) from temperature (°C) and relative humidity (%).

This function is under development.

Usage

```
calcFP(Temp, RH)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)

Details

Formula coefficients from Arden Buck equation (1981, 1996) saturation vapor pressure over ice.

- a = 6.1115
- b = 23.036
- c = 279.82
- d = 333.7

Value

Tf, Frost Point (°Celsius)

Examples

```
calcFP(20, 50)
calcFP(0, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(FrostPoint = calcFP(Temp, RH))
```

---

calcFtoC *Convert temperature (F) to temperature (C)*

Description

Convert temperature in Fahrenheit to temperature in Celsius

Usage

```
calcFtoC(TempF)
```

Arguments

TempF	Temperature (Fahrenheit)

Value

TempC Temperature (Celsius)

Examples

```
calcFtoC(32)
calcFtoC(68)

# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(TempC = calcFtoC((Temp * 9/5) + 32))
```

`calcHR` *Calculate Humidity Ratio*

Description

Function to calculate humidity ratio (g/kg) from temperature (°C) and relative humidity (%).

Humidity ratio is the mass of water vapor present in a given volume of air relative to the mass of dry air. Also known as "moisture content".

Function uses `calcMR`

Usage

```
calcHR(Temp, RH, P_atm = 1013.25, B = 621.9907, ...)
```

Arguments

`Temp`	Temperature (°Celsius)
`RH`	Relative Humidity (0-100%)
`P_atm`	Atmospheric pressure = 1013.25 (hPa)
`B`	B = 621.9907 g/kg for air
`...`	Additional arguments to supply to `calcPws` and `calcMR`

Value

HR Humidity ratio (g/kg)

Note

This function requires the `calcMR` function to be available in the environment.

See Also

`calcMR` for calculating mixing ratio

`calcAD` for calculating air density

`calcPw` for calculating water vapour pressure

`calcPws` for calculating water vapour saturation pressure

Examples

```
calcHR(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)
```

```
mydata |> dplyr::mutate(HumidityRatio = calcHR(Temp, RH))
```

`calcLM` *Life-time multiplier for chemical degradation*

Description

Function to calculate lifetime multiplier from temperature and relative humidity.

The lifetime multiplier calculates the expected lifetime of an object for a given point relative to to the lifetime at 20°C and 50%RH. This can used to then average over the length of the dataset.

The lifetime multiplier gives an indication of the speed of natural decay of an object. It expresses an expected lifetime of an object compared to the expected lifetime of the same object at 20°C and 50% RH. This means that if the result = 1, the expected lifetime for your object is 'good'. The closer you go to 0, the less suited your environment is. The result is both expressed numerically and over time, which also gives an idea about the period over the year when the object suffers most. The data is based on experiments on paper, synthetic films and dyes.

Usage

```
calcLM(Temp, RH, EA = 100)
```

Arguments

`Temp`	Temperature (Celsius)
`RH`	Relative Humidity (0-100%)
`EA`	Activation Energy (J/mol). 100 J/mol for cellulosic (paper) or 70 J/mol yellowing varnish

Details

Based on the experiments of the rate of decay of paper, films and dyes. Activation energy, Ea = 100 J/mol (degradation of cellulose - paper), 70 J/mol (yellowing of varnish - furniture, painting, sculpture).

Gas constant, R = 8.314 J/K.mol

$$LM = \left(\frac{50\%RH}{RH}\right)^{1.3} .e\left(\frac{E_a}{R}.\left(\frac{1}{T_K} - \frac{1}{293}\right)\right)$$

Value

Lifetime multiplier

References

Michalski, S., 'Double the life for each five-degree drop, more than double the life for each halving of relative humidity', in Preprints of the 13th IcOM-cc Triennial Meeting in rio de Janeiro (22–27 September 2002), ed. r. Vontobel, James & James, London (2002) Vol. I 66–72.

Martens Marco, 2012: Climate Risk Assessment in Museums (Thesis, Tue).

Examples

```
calcLM(20, 50)

calcLM(20, 50, EA = 70)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(LifeTime = calcLM(Temp, RH))

mydata |>
  dplyr::mutate(LM = calcLM(Temp, RH)) |>
   dplyr::summarise(LM_avg = mean(LM, na.rm = TRUE))
```

`calcMould_VTT` *Calculate Mould Growth Index (VTT model)*

Description

This function calculates the mould growth index on wooden materials based on temperature, relative humidity, and other factors. It implements the mathematical model developed by Hukka and Viitanen, which predicts mould growth under varying environmental conditions.

Usage

```
calcMould_VTT(
  Temp,
  RH,
  M_prev = 0,
  sensitivity = "very",
  wood = 0,
  surface = 0
)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)
M_prev	The previous mould index value (default is 0).
sensitivity	The sensitivity level of the material to mould growth. Options are 'very', 'sensitive', 'medium', or 'resistant'. Default is 'very'.
wood	The wood species; 0 for pine and 1 for spruce. Default is 0.
surface	The surface quality; 0 for resawn kiln dried timber and 1 for timber dried under normal kiln drying process. Default is 0 (worst case).

Details

Senstivity is related to the material surface, mould will grow on. Options in function avaiable are:

- 'very' sensitive materials include pine and sapwood.
- 'sensitive' materials include glued wooden boards, PUR with paper surface, spruce
- 'medium' resistant materials include concrete, glass wool, polyester wool
- 'resistant' materials include PUR polished surface

Value

M Mould growth index

- 0 = No mould growth
- 1 = Small amounts of mould growth on surface visible under microscope
- 2 = Several local mould growth colonies on surface visible under microscope
- 3 = Visual findings of mould on surface <10% coverage or 50% coverage under microsocpe
- 4 = Visual findings of mould on surface 10-50% coverage or >50% coverage under microscope
- 5 = Plenty of growth on surface >50% visual coverage
- 6 = Heavy and tight growth, coverage almost 100%

References

Hukka, A., Viitanen, H. A mathematical model of mould growth on wooden material. Wood Science and Technology 33, 475–485 (1999). https://doi.org/10.1007/s002260050131

Viitanen, Hannu, and Tuomo Ojanen. "Improved model to predict mold growth in building materials." Thermal Performance of the Exterior Envelopes of Whole Buildings X–Proceedings CD (2007): 2-7.

Examples

```
calcMould_VTT(Temp = 25, RH = 85)

calcMould_VTT(Temp = 18, RH = 70, M_prev = 2, sensitivity = "medium", wood = 1, surface = 1)

# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |>
   dplyr::mutate(
      MouldIndex = calcMould_VTT(Temp, RH),
      MouldIndex_sensitve = calcMould_VTT(Temp, RH, sensitivity = "sensitive")
   )
```

---

calcMould_Zeng *Calculate Mould Growth Rate Limits (Zeng et al.)*

Description

This function calculates the Lowest Isoline for Mould (LIM) based on temperature and relative humidity, using the model developed by Zeng et al. (2023).

The LIM is the lowest envelope of the temperature and humidity isoline at a certain mould growth rate (u). LIM0 is the critical value for mould growth, if the humidity is kept below the critcal value, at a given temperature, then there is no risk of mould growth.

Usage

```
calcMould_Zeng(Temp, RH, LIM = 0, label = FALSE)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)
LIM	The specific LIM value to calculate. Must be one of 0, 0.1, 0.5, 1, 2, 3, or 4. Default is 0.
label	Logical. If TRUE, returns a descriptive label instead of a numeric value. Default is FALSE.

Details

The function calculates LIM values for mould genera including Cladosporium, Penicillium, and Aspergillus. LIM values represent different mould growth rates:

- LIM0: Low limit of mould growth
- LIM0.1: 0.1 mm/day growth rate
- LIM0.5: 0.5 mm/day growth rate
- LIM1: 1 mm/day growth rate
- LIM2: 2 mm/day growth rate
- LIM3: 3 mm/day growth rate
- LIM4: 4 mm/day growth rate
- Above LIM4: Greater than 4 mm/day growth rate (9 mm/day theorectical maximum)

Value

If label is FALSE, returns the calculated LIM value as Relative Humidity (0-100%). If label is TRUE, returns a character string describing the mould growth rate category.

References

Zeng L, Chen Y, Ma M, et al. Prediction of mould growth rate within building envelopes: development and validation of an improved model. Building Services Engineering Research and Technology. 2023;44(1):63-79. doi:10.1177/01436244221137846

Sautour M, Dantigny P, Divies C, Bensoussan M. A temperature-type model for describing the relationship between fungal growth and water activity. Int J Food Microbiol. 2001 Jul 20;67(1-2):63-9. doi: 10.1016/s0168-1605(01)00471-8. PMID: 11482570.

Examples

```
calcMould_Zeng(20, 75)
calcMould_Zeng(20, 75, LIM = 0)
calcMould_Zeng(20, 75, label = TRUE)

calcMould_Zeng(20, 85)
calcMould_Zeng(20, 85, LIM = 2)
calcMould_Zeng(20, 85, label = TRUE)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |>
   dplyr::mutate(
      RH_LIM0 = calcMould_Zeng(Temp, RH),
      RH_LIM1 = calcMould_Zeng(Temp, RH, LIM = 1),
      LIM = calcMould_Zeng(Temp, RH, label = TRUE)
   )
```

`calcMR` *Calculate Mixing Ratio*

Description

Function to calculate mixing ratio (g/kg) from temperature (°C) and relative humidity (%).

Mixing Ratio is the mass of water vapor present in a given volume of air relative to the mass of dry air.

Usage

```
calcMR(Temp, RH, P_atm = 1013.25, B = 621.9907, ...)
```

Arguments

`Temp`	Temperature (°Celsius)
`RH`	Relative Humidity (0-100%)
`P_atm`	Atmospheric pressure = 1013.25 (hPa)
`B`	B = 621.9907 g/kg for air
`...`	Additional arguments to supply to `calcPws`

Details

X Mixing ratio (mass of water vapour / mass of dry gas)

Pw = Pws(40°C) = 73.75 hPa

X = 621.9907 x 73.75 / (998 - 73.75) = 49.63 g/kg

Value

X Mixing ratio, mass of water vapour / mass of dry gas (g/kg)

See Also

`calcMR` for calculating mixing ratio

`calcAD` for calculating air density

`calcPw` for calculating water vapour pressure

`calcPws` for calculating water vapour saturation pressure

Examples

```
calcMR(20, 50)

# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(MixingRatio = calcMR(Temp, RH))
```

---

calcPI *Calculate Preservation Index*

Description

Calculates the Preservation Index (PI) to estimate the natural decay speed of objects. Uses acetate film as a reference for early warning of chemical deterioration in organic and synthetic objects. Results are in years for each data point, showing periods of higher and lower risk. Model based on acetate films or similarly unstable objects under specific temperature and relative humidity conditions. The model is less reliable at high and low RH values.

Usage

```
calcPI(Temp, RH, EA = 90300)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)
EA	Activation Energy (J/mol). Default is 90300 J/mol for cellulose acetate film

Details

The formula is based on Arrhenius equation (for molecular energy) and an equivalent for E (best fit to the cellulose triacetate deterioration data). The other parameters are integrated to mimic the results from the experiments. The result is an average chemical lifetime at one point in time of chemically unstable object (based on experiments on acetate film). This means it is the expected lifetime for a specific T, RH and theoretical object if this remains stable (no fluctuations). The chosen activation energy (Ea) has a larger impact at low temperature.

Developed by the Image Permance Institute, the model is based on the chemical degradation of cellulose acetate (Reilly et al., 1995):

- Rate, k = [RH%] × 5.9 × 10^12 × exp(-90300 / (8.314 × TempK))

- Preservation Index, PI = 1/k

Value

PI Preservation Index, the expected lifetime (1/rate,k)

Note

This metric is an early version of a lifetime multiplier based on chemical deterioration of acetate film. This last object is naturally relatively unstable and there lies the biggest difference with other chemical metrics together with the fact that it is not relative to the lifetime of the object. All lifetime multipliers give similar results between 20% and 60% RH. The results at very low and high RH can be unreliable.

References

Reilly, James M. New Tools for Preservation: Assessing Long-Term Environmental Effects on Library and Archives Collections. Commission on Preservation and Access, 1400 16th Street, NW, Suite 740, Washington, DC 20036-2217, 1995.

Padfield, T. 2004. The Preservation Index and The Time Weighted Preservation Index. https://www.conservationphysics.org/t

Activation Energy: ASHRAE, 2011.

Image Permanence Institute, eClimateNotebook

Examples

```
calcPI(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(PI = calcPI(Temp, RH))
```

---

calcPw *Calculate Water Vapour Pressure*

Description

Function to calculate water vapour pressure (hPa) from temperature (°C) and relative humidity (%).

Water vapour pressure is the pressure exerted by water vapour in a gas.

Usage

```
calcPw(Temp, RH, ...)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)
...	Additional arguments to supply to calcPws

Details

Different formulations for calculating water vapour pressure are available:

- Arden Buck equation ("Buck")
- International Association for the Properties of Water and Steam ("IAPWS")
- August-Roche-Magnus approximation ("Magnus")
- VAISALA humidity conversion formula ("VAISALA")

The water vapor pressure (P_w) is calculated using the following equation:

$$P_w = \frac{P_{ws}\,(Temp) \times RH}{100}$$

Where:

- P_ws is the saturation vapor pressure using calcPws.
- RH is the relative humidity in percent.
- Temp is the temperature in degrees Celsius.

Value

Pw, Water Vapour Pressure (hPa)

Note

See Wikipedia for a discussion of the accuarcy of each approach: https://en.wikipedia.org/wiki/Vapour_pressure_of_water

References

Wagner, W., & Pru\ß, A. (2002). The IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use. Journal of Physical and Chemical Reference Data, 31(2), 387-535.

Alduchov, O. A., and R. E. Eskridge, 1996: Improved Magnus' form approximation of saturation vapor pressure. J. Appl. Meteor., 35, 601-609.

Buck, A. L., 1981: New Equations for Computing Vapor Pressure and Enhancement Factor. J. Appl. Meteor. Climatol., 20, 1527–1532, https://doi.org/10.1175/1520-0450(1981)020<1527:NEFCVP>2.0.CO;2.

Buck (1996), Buck (1996), Buck Research CR-1A User's Manual, Appendix 1.

VAISALA. Humidity Conversions: Formulas and methods for calculating humidity parameters. Ref. B210973EN-O

See Also

calcMR for calculating mixing ratio

calcAD for calculating air density

calcPw for calculating water vapour pressure

calcPws for calculating water vapour saturation pressure

Examples

```
calcPw(20, 50)

# Calculate relative humidity at 50%RH
calcPw(20, 50) / calcPws(20) * 100


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(Pw = calcPw(Temp, RH))

mydata |> dplyr::mutate(Buck = calcPw(Temp, RH, method = "Buck"),
                              IAPWS = calcPw(Temp, RH, method = "IAPWS"),
                              Magnus = calcPw(Temp, RH, method = "Magnus"),
                              VAISALA = calcPw(Temp, RH, method = "VAISALA"))
```

---

calcPws *Calculate Water Vapour Saturation Pressure*

Description

Function to calculate water vapour saturation pressure (hPa) from temperature (°C) using the International Association for the Properties of Water and Steam (IAPWS as default), Arden Buck equation (Buck), August-Roche-Magnus approximation (Magnus) or VAISALA conversion formula.

Water vapour saturation pressure is the maximum partial pressure of water vapour that can be present in gas at a given temperature.

Usage

```
calcPws(
  Temp,
  P_atm = 1013.25,
  method = c("Buck", "IAPWS", "Magnus", "VAISALA")
)
```

Arguments

Temp	Temperature (°Celsius)
P_atm	Atmospheric pressure = 1013.25 (hPa)
method	Character. Method to use for calculation. Options are "Buck" (default), "IAPWS", "Magnus" or "VAISALA".

Details

Different formulations for calculating water vapour pressure are available:

- Arden Buck equation ("Buck")
- International Association for the Properties of Water and Steam ("IAPWS")
- August-Roche-Magnus approximation ("Magnus")
- VAISALA humidity conversion formula ("VAISALA")

Value

Pws, Saturation vapor pressure (hPa)

Note

See Wikipedia for a discussion of the accuarcy of each approach: https://en.wikipedia.org/wiki/Vapour_pressure_of_water

If lower accuracy or a limited temperature range can be tolerated a simpler formula can be used for the water vapour saturation pressure over water (and over ice):

Pws = 6.116441 x 10^((7.591386 x Temp) / (Temp + 240.7263))

References

Wagner, W., & Pru\ß, A. (2002). The IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use. Journal of Physical and Chemical Reference Data, 31(2), 387-535.

Alduchov, O. A., and R. E. Eskridge, 1996: Improved Magnus' form approximation of saturation vapor pressure. J. Appl. Meteor., 35, 601-609.

Buck, A. L., 1981: New Equations for Computing Vapor Pressure and Enhancement Factor. J. Appl. Meteor. Climatol., 20, 1527–1532, https://doi.org/10.1175/1520-0450(1981)020<1527:NEFCVP>2.0.CO;2.

Buck (1996), Buck (1996), Buck Research CR-1A User's Manual, Appendix 1.

VAISALA. Humidity Conversions: Formulas and methods for calculating humidity parameters. Ref. B210973EN-O

See Also

calcMR for calculating mixing ratio

calcAD for calculating air density

calcPw for calculating water vapour pressure

calcPws for calculating water vapour saturation pressure

Examples

```
calcPws(20)
calcPws(20, method = "Buck")
calcPws(20, method = "IAPWS")
calcPws(20, method = "Magnus")
calcPws(20, method = "VAISALA")

# Check of calculations of relative humidity at 50%RH
calcPw(20, 50, method = "Buck") / calcPws(20, method = "Buck") * 100
calcPw(20, 50, method = "IAPWS") / calcPws(20, method = "IAPWS") * 100
calcPw(20, 50, method = "Magnus") / calcPws(20, method = "Magnus") * 100
calcPw(20, 50, method = "VAISALA") / calcPws(20, method = "VAISALA") * 100


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(Pws = calcPws(Temp))

mydata |> dplyr::mutate(Buck = calcPws(Temp, method = "Buck"),
                            IAPWS = calcPws(Temp, method = "IAPWS"),
                            Magnus = calcPws(Temp, method = "Magnus"),
                            VAISALA = calcPws(Temp, method = "VAISALA"))
```

---

calcRH_AH *Calculate Relative Humidity from temperature and absolute humidity*

Description

Function to calculate relative humidity (%) from temperature (°C) and absolute humidity (g/m^3)

Usage

```
calcRH_AH(Temp, Abs, P_atm = 1013.25)
```

Arguments

Temp	Temperature (°Celsius)
Abs	Absolute Humidity (g/m³)
P_atm	Atmospheric pressure = 1013.25 (hPa)

Value

Relative Humidity (0-100%)

References

Buck, A. L. (1981). New equations for computing vapor pressure and enhancement factor. Journal of Applied Meteorology, 20(12), 1527-1532.

See Also

calcAH for calculating absolute humidity

calcTemp for calculating temperature

calcRH_DP for calculating relative humidity from dew point

calcDP for calculating dew point

Examples

```
# Calculate RH for temperature of 20°C and absolute humidity of 8.645471 g/m³
calcRH_AH(20, 8.630534)

calcRH_AH(20, calcAH(20, 50))


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(Abs = calcAH(Temp, RH), RH2 = calcRH_AH(Temp, Abs))
```

---

calcRH_DP *Calculate Relative Humidity from temperature and dew point*

Description

Function to calculate relative humidity (%) from temperature (°C) and dew point (°C)

Usage

```
calcRH_DP(Temp, DewP, method = c("Magnus", "Buck"))
```

Arguments

Temp	Temperature (°Celsius)
DewP	Td (DP), Dew Point (°Celsius)
method	Calculation method: either "Magnus" or "Buck". Defaults to "Magnus".

Details

This function supports two methods for relative humidity calculation:

- "Magnus" (default): Uses the August-Roche-Magnus approximation, valid for 0°C < Temp < 60°C and 1% < RH < 100%.
- "Buck": Uses the Arden Buck equation with Bögel modification, valid for -30°C < Temp < 60°C and 1% < RH < 100%.

The methods calculate temperature based on vapor pressure and saturation vapour pressure relationships. The Magnus method is chosen as the default because it is more stable when used with the calcDP and calcTemp functions.

Value

Relative Humidity (0-100%)

References

Alduchov, O. A., and R. E. Eskridge, 1996: Improved Magnus' form approximation of saturation vapor pressure. J. Appl. Meteor., 35, 601–609

Buck, A. L., 1981: New Equations for Computing Vapor Pressure and Enhancement Factor. J. Appl. Meteor. Climatol., 20, 1527–1532, https://doi.org/10.1175/1520-0450(1981)020<1527:NEFCVP>2.0.CO;2.

Buck (1996), Buck (1996), Buck Research CR-1A User's Manual, Appendix 1.

https://bmcnoldy.earth.miami.edu/Humidity.html

See Also

calcTemp for calculating temperature

calcDP for calculating dew point

calcRH_AH for calculating relative humidity from absolute humidity

calcRH_DP for calculating relative humidity from dew point

Examples

```
# RH at air tempertaure of 20°C and dew point of 15°C
calcRH_DP(20, 15)
calcRH_DP(20, 15, method = "Buck")

calcRH_DP(20, calcDP(20, 50))

calcRH_DP(20, calcDP(20, 50, method = "Buck"), method = "Buck")

# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |>
  dplyr::mutate(
```

```
DewPoint = calcDP(Temp, RH),
RH_default = calcRH_DP(Temp, DewPoint),
RH_Buck = calcRH_DP(Temp, DewPoint, method = "Buck"))
```

`calcSensibleHeating` *Calculate Sensible Heating*

Description

This function calculates sensible heating power.

Sensible heat is the energy that causes an object's temperature to change without altering its phase, also known as "dry" heat which you can feel.

Usage

```
calcSensibleHeating(Temp1, Temp2, RH = 50, volumeFlowRate)
```

Arguments

`Temp1`	Initial Temperature (°Celsius)
`Temp2`	Final Temperature (°Celsius)
`RH`	Initial Relative Humidity (0-100%). Optional, default is 50%.
`volumeFlowRate`	Volume flow rate of air in cubic meters per second (m³/s)

Value

Sensible heat in kilowatts (kW)

See Also

`calcAD`

Examples

```
calcSensibleHeating(20, 25, 50, 0.5)

calcSensibleHeating(20, 25, 60, 0.5)
```

calcSensibleHeatRatio *Calculate Sensible Heat Ratio (SHR)*

Description

This function calculates the Sensible Heat Ratio (SHR) using the sensible and total heating values. Sensible heat ratio is the ratio of sensible heat to total heat.

Usage

```
calcSensibleHeatRatio(Temp1, Temp2, RH1, RH2, volumeFlowRate)
```

Arguments

Temp1	Initial Temperature (°Celsius)
Temp2	Final Temperature (°Celsius)
RH1	Initial Relative Humidity (0-100%)
RH2	Final Relative Humidity (0-100%)
volumeFlowRate	Volume flow rate of air in cubic meters per second (m³/s)

Value

SHR Sensible Heat Ratio (0-100%)

See Also

calcSensibleHeating, calcTotalHeating

Examples

```
calcSensibleHeatRatio(20, 25, 50, 30, 0.5)
```

calcSH *Calculate Specific Humidity*

Description

Function to calculate the specific humidity (g/kg) from temperature (°C) and relative humidity (%). Specific humidity is the ratio of the mass of water vapor to the mass of air.

Function uses calcMR

Usage

```
calcSH(Temp, RH, P_atm = 1013.25, B = 621.9907, ...)
```

Arguments

Temp	Temperature (°Celsius)
RH	Relative Humidity (0-100%)
P_atm	Atmospheric pressure = 1013.25 (hPa)
B	B = 621.9907 g/kg for air
...	Additional arguments to supply to calcPws and calcMR

Value

SH Specific Humidity (g/kg)

Note

This function requires the calcMR function to be available in the environment.

References

Wallace, J.M. and Hobbs, P.V. (2006). Atmospheric Science: An Introductory Survey. Academic Press, 2nd edition.

See Also

calcAD for calculating air density

calcAH for calculating absolute humidity

calcPw for calculating water vapour pressure

calcPws for calculating water vapour saturation pressure

Examples

```
calcSH(20, 50)


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |> dplyr::mutate(SpecificHumidity = calcSH(Temp, RH))
```

`calcTemp` *Calculate Temperature from relative humidity and dew point*

Description

This function calculates the temperature (°C) from relative humidity (%) and dew point temperature (°C).

Usage

```
calcTemp(RH, DewP, method = c("Magnus", "Buck"))
```

Arguments

`RH`	Relative Humidity (0-100%)
`DewP`	Td (DP), Dew Point (°Celsius)
`method`	Calculation method: either `"Magnus"` or `"Buck"`. Defaults to `"Magnus"`.

Details

This function supports two methods for temperature calculation:

- `"Magnus"` (default): Uses the August-Roche-Magnus approximation, valid for 0°C < Temp < 60°C and 1% < RH < 100%.
- `"Buck"`: Uses the Arden Buck equation with Bögel modification, valid for -30°C < Temp < 60°C and 1% < RH < 100%.

The methods calculate temperature based on vapor pressure and saturation vapour pressure relationships. The Magnus method is chosen as the default because it is more stable when used with the `calcDP` and `calcRH_DP` functions.

Value

Temp, Temperature (°Celsius)

References

Alduchov, O. A., and R. E. Eskridge, 1996: Improved Magnus' form approximation of saturation vapor pressure. J. Appl. Meteor., 35, 601–609

Buck, A. L., 1981: New Equations for Computing Vapor Pressure and Enhancement Factor. J. Appl. Meteor. Climatol., 20, 1527–1532, https://doi.org/10.1175/1520-0450(1981)020<1527:NEFCVP>2.0.CO;2.

Buck (1996), Buck (1996), Buck Research CR-1A User's Manual, Appendix 1.

https://bmcnoldy.earth.miami.edu/Humidity.html

See Also

calcTemp for calculating temperature

calcDP for calculating dew point

calcRH_DP for calculating relative humidity from dew point

calcRH_AH for calculating relative humidity from absolute humidity

Examples

```
# Calculate temperature for RH 50\% and Dew Point 15°C using Magnus method
calcTemp(50, 15)

# Using Buck method
calcTemp(50, 15, method = "Buck")

calcTemp(50, calcDP(20, 50))


# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 5)

mydata |>
  dplyr::mutate(
    DewPoint = calcDP(Temp, RH),
    Temp_default = calcTemp(RH, DewPoint),
    Temp_Buck = calcTemp(RH, DewPoint))
```

---

calcTotalHeating *Calculate Total Heating*

Description

This function calculates total heating power.

Total heating power is the sum of sensible (felt) heat and latent (hidden) heat.

Usage

```
calcTotalHeating(Temp1, Temp2, RH1, RH2, volumeFlowRate)
```

Arguments

Temp1	Initial Temperature (°Celsius)
Temp2	Final Temperature (°Celsius)
RH1	Initial Relative Humidity (0-100%)
RH2	Final Relative Humidity (0-100%)
volumeFlowRate	Volume flow rate of air in cubic meters per second (m³/s)

Value

Total Heating in kilowatts (kW)

See Also

calcAD, calcEnthalpy

Examples

```
calcTotalHeating(20, 25, 50, 30, 0.5)
```

`data_file_path` *Return the file path to data files*

Description

Access mydata.xlsx and other files in inst/extdata folder

Usage

```
data_file_path(path = NULL)
```

Arguments

`path`	Name of file in quotes with extension, e.g. "mydata.xlsx"

Value

String of the path to the specified file

Examples

```
data_file_path()

data_file_path("mydata.xlsx")
```

graph_psychrometric *Create a Psychrometric Chart*

Description

This function generates a psychrometric chart based on input temperature and relative humidity data.

Various psychrometric functions can be used for the y-axis.

- calcHR: Humidity Ratio (g/kg)
- calcMR: Mixing Ratio (g/kg)
- calcAH: Absolute Humidity (g/m^3)
- calcSH: Specific Humidity (g/kg)
- calcAD: Air Density (kg/m^3)
- calcDP: Dew Point (°C)
- calcFP: Frost Point (°C)
- calcEnthalpy: Enthalpy (kJ/kg)
- calcPws: Saturation vapor pressure (hPa)
- calcPw: Water Vapour Pressure (hPa)
- calcPI: Preservation Index
- calcLM: Lifetime
- calcEMC_wood: Equilibrium Moisture Content (wood)

Usage

```
graph_psychrometric(
  mydata,
  Temp = "Temp",
  RH = "RH",
  data_colour = "RH",
  data_alpha = 0.5,
  LowT = 16,
  HighT = 25,
  LowRH = 40,
  HighRH = 60,
  Temp_range = c(0, 40),
  y_func = calcMR,
  ...
)
```

Arguments

mydata	A data frame containing temperature and relative humidity data.
Temp	Column name in mydata for temperature values.
RH	Column name in mydata for relative humidity values.
data_colour	Column name to use to colour the data points. Default is "RH".
data_alpha	Value to supply for make points more or less transparent. Default is 0.5.
LowT	Numeric value for lower temperature limit of the target range. Default is 16°C.
HighT	Numeric value for upper temperature limit of the target range. Default is 25°C.
LowRH	Numeric value for lower relative humidity limit of the target range. Default is 40%.
HighRH	Numeric value for upper relative humidity limit of the target range. Default is 60%.
Temp_range	Numeric vector of length 2 specifying the overall temperature range for the chart. Default is c(0, 40).
y_func	Function to calculate y-axis values. See above for options, default is calcMR (mixing ratio).
...	Additional arguments passed to y_func.

Value

A ggplot object representing the psychrometric chart.

Examples

```
# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 100)

# Basic usage with default settings
graph_psychrometric(mydata, Temp, RH)

# Custom temperature and humidity ranges
graph_psychrometric(mydata, Temp, RH, LowT = 8, HighT = 28, LowRH = 30, HighRH = 70)

# Using a different psychrometric function (e.g., Absolute Humidity)
graph_psychrometric(mydata, Temp, RH, y_func = calcAH)

# Adjusting the overall temperature range of the chart
graph_psychrometric(mydata, Temp, RH, Temp_range = c(12, 30))
```

graph_TRH *Function for graphing temperature and humidity data*

Description

Use this tool to produce a simple temperature and humidity plot

Usage

```
graph_TRH(mydata, Date = "Date", Temp = "Temp", RH = "RH")
```

Arguments

mydata	A data frame containing the date, temperature, and relative humidity data.
Date	The name of the column in mydata containing date information.
Temp	The name of the column in mydata containing temperature data.
RH	The name of the column in mydata containing relative humidity data.

Value

A ggplot graph of temperature and relative humidity.

Examples

```
# mydata file
filepath <- data_file_path("mydata.xlsx")
mydata <- readxl::read_excel(filepath, sheet = "mydata", n_max = 1000)

graph_TRH(mydata)
```

mydata *Climate dataset to demonstrate functions*

Description

A climate dataset for use to demonstrate how the functions work.

Usage

```
mydata
```

Format

A data frame with 35,136 rows and 5 columns:

Site Site location name

Sensor Sensor name, unique to the site

Date Date is ISOdate time format

Temp, RH Temperature (C) and relative humidity (%) ...

Source

Climate

`run_ConSciR_app` *Run ConSciR Shiny Application*

Description

Run ConSciR Shiny Application

Usage

```
run_ConSciR_app()
```

Value

Shiny object

Examples

```
if(interactive()) {
    run_ConSciR_app()
}
```

run_Mould_app *Run ConSciR Mould Application*

Description

Shiny application to upload data to estimate mould growth predictions.

CSV or Excel formatted data with "Temp" and "RH" columns can be uploaded to the application.

Functions available:

- calcMould_Zeng: Mould Growth Rate Limits
- calcMould_VTT: Mould Growth Index (VTT model)

Usage

```
run_Mould_app()
```

Value

Shiny object

Examples

```
if(interactive()) {
    run_Mould_app()
}
```

run_Psychrometric_app *Run ConSciR Psychrometric and Graphing Application*

Description

Shiny application to upload data to a psychrometric chart. Also includes graphs for temperature and humidity - line plot with limits shaded and a bivariate plot with box showing limits.

CSV or Excel formatted data with "Temp" and "RH" columns can be uploaded to the application.

Use the sliders and functions to set the limits and parameters to be used.

Functions available:

- calcHR: Humidity Ratio (g/kg)
- calcMR: Mixing Ratio (g/kg)
- calcAH: Absolute Humidity (g/m^3)
- calcSH: Specific Humidity (g/kg)

- calcAD: Air Density (kg/m^3)
- calcDP: Dew Point (°C)
- calcEnthalpy: Enthalpy (kJ/kg)
- calcPws: Saturation vapor pressure (hPa)
- calcPw: Water Vapour Pressure (hPa)
- calcPI: Preservation Index
- calcLM: Lifetime
- calcEMC_wood: Equilibrium Moisture Content (wood)

Usage

```
run_Psychrometric_app()
```

Value

Shiny object

Examples

```
if(interactive()) {
    run_Psychrometric_app()
}
```

`run_SilicaGel_app` *Run ConSciR Silica Gel Calculator Application*

Description

Shiny application to calculate the amount of silica gel required. Temperature and humidity data from the proposed location, case dimensions, and air exchange rate (AER) if known, as inputs.

CSV or Excel data with "Date", "Temp" and "RH" columns can be uploaded to the application.

Usage

```
run_SilicaGel_app()
```

Value

Shiny object

Examples

```
if(interactive()) {
    run_SilicaGel_app()
}
```

run_TidyData_app *Run ConSciR Tidying Data Application*

Description

Shiny application to help with tidying data.

CSV or Excel formatted data can be uploaded to the application.

Usage

```
run_TidyData_app()
```

Value

Shiny object

Examples

```
if(interactive()) {
    run_TidyData_app()
}
```

run_TRHbivariate_app *Run ConSciR Temperature and Humidity Application*

Description

Shiny application to plot temperature and humidity data on a bivariate chart. A summary of the data is also produced by specifying a temperature and humidity box.

CSV or Excel formatted data with "Temp" and "RH" columns can be uploaded to the application.

Usage

```
run_TRHbivariate_app()
```

Value

Shiny object

Examples

```
if(interactive()) {
    run_TRHbivariate_app()
}
```

shiny_dataUploadServer *Shiny Module Server for Data Upload and Processing*

Description

This function creates a Shiny module server for uploading CSV or Excel files, processing the data, and returning a tidied dataset.

Usage

```
shiny_dataUploadServer(id)
```

Arguments

id	A character string that corresponds to the ID used in the UI function for this module.

Value

A reactive expression containing the tidied data frame with the following columns:

- Date: Date and time, floored to the hour
- Sensor: Sensor identifier
- Site: Site identifier
- Temp: Median average temperature for each hour
- RH: Median average relative humidity for each hour

Examples

```
if(interactive()) {

# In a Shiny app:
ui <- fluidPage(
  shiny_dataUploadUI("dataUpload")
)

server <- function(input, output, session) {
  data <- shiny_dataUploadServer("dataUpload")
}

}
```

shiny_dataUploadUI *UI Module for Data Upload in Shiny*

Description

This function creates a Shiny UI module for uploading data files. It provides a file input interface that can be integrated into a larger Shiny application.

Usage

```
shiny_dataUploadUI(id)
```

Arguments

id	A character string that defines the namespace for the module's UI elements.

Value

A 'tagList' containing a 'uiOutput' for file upload. The specific elements of this output (such as file input and upload button) are defined in the corresponding server function.

Examples

```
if(interactive()) {

# In a Shiny app:
ui <- fluidPage(
  shiny_dataUploadUI("dataUpload")
)

server <- function(input, output, session) {
  data <- shiny_dataUploadServer("dataUpload")
}

}
```

tidy_Hanwell *Tidy Hanwell EMS Data (Min-Max report)*

Description

This function reads and processes Hanwell Environmental Monitoring System (EMS) data, transforming it into a tidy format suitable for analysis.

Usage

```
tidy_Hanwell(EMS_MinMax_datapath)
```

Arguments

EMS_MinMax_datapath
: EMS_MinMax_datapath Character string specifying the file path to the Hanwell EMS data file.

Value

A tibble containing the tidied Hanwell EMS data with the following columns:

- Date Time POSIXct datetime of the measurement
- Sensor Character string identifying the sensor
- TempMin Numeric minimum temperature (°C)
- TempMax Numeric maximum temperature (°C)
- Temp Numeric average temperature (°C)
- RHMin Numeric minimum relative humidity (
- RHMax Numeric maximum relative humidity (
- RH Numeric average relative humidity (
- Date Date of the measurement (duplicate of Date Time)

Examples

```
# Example usage: hanwell_data <- tidy_Hanwell("path/to/your/EMS_MinMax_data.csv")
```

---

tidy_Meaco *Tidy Meaco sensor data*

Description

This function takes raw Meaco sensor data and performs several cleaning and processing steps:

- Filters out rows with missing dates
- Renames column names for consistency
- Converts temperature and relative humidity to numeric
- Rounds dates down to the nearest hour
- Calculates hourly averages for temperature and relative humidity
- Pads the data to ensure hourly intervals using padr package
- Filters out unrealistic temperature and humidity values (outside -50°C to 50°C and 0 to 100%RH)

Usage

```
tidy_Meaco(
  mydata,
  Site_col = "RECEIVER",
  Sensor_col = "TRANSMITTER",
  Date_col = "DATE",
  Temp_col = "TEMPERATURE",
  RH_col = "HUMIDITY"
)
```

Arguments

mydata	A data frame containing raw Meaco sensor data with columns RECEIVER, TRANSMITTER, DATE, TEMPERATURE, and HUMIDITY
Site_col	A string specifying the name of the column in ‘mydata‘ that contains location information. Default is "RECEIVER".
Sensor_col	A string specifying the name of the column in ‘mydata‘ that contains sensor information. Default is "TRANSMITTER".
Date_col	A string specifying the name of the column in ‘mydata‘ that contains date information. Default is "DATE".
Temp_col	A string specifying the name of the column in ‘mydata‘ that contains temperature data. Default is "TEMPERATURE".
RH_col	A string specifying the name of the column in ‘mydata‘ that contains relative humidity data. Default is "HUMIDITY".

Value

A tidied data frame with columns Site, Sensor, Date, Temp, and RH

Examples

```
# Example usage: meaco_data <- tidy_Meaco("path/to/your/meaco_data.csv")
```

---

tidy_TRHdata *Tidy and Process Temperature and Relative Humidity data*

Description

This function tidies and processes temperature, relative humidity, and date data from a given dataset.

It filters out rows with missing date values, renames columns, converts temperature and humidity to numeric types, and groups the data by site, sensor, and date. The function also pads the data to ensure hourly intervals.

- Filters out rows with missing dates
- Renames columns for consistency
- Converts temperature and relative humidity to numeric
- Rounds dates down to the nearest hour
- Calculates hourly averages for temperature and relative humidity
- Pads the data to ensure hourly intervals
- Filters out implausible temperature and humidity values

Usage

```
tidy_TRHdata(
  mydata,
  Site_col = "Site",
  Sensor_col = "Sensor",
  Date_col = "Date",
  Temp_col = "Temp",
  RH_col = "RH"
)
```

Arguments

mydata	A data frame containing the raw TRH data. This should include columns for site, sensor, date, temperature, and relative humidity.
Site_col	A string specifying the name of the column in ‘mydata‘ that contains location information. Default is "Site".
Sensor_col	A string specifying the name of the column in ‘mydata‘ that contains sensor information. Default is "Sensor".
Date_col	A string specifying the name of the column in ‘mydata‘ that contains date information. Default is "Date".
Temp_col	A string specifying the name of the column in ‘mydata‘ that contains temperature data. Default is "Temp".
RH_col	A string specifying the name of the column in ‘mydata‘ that contains relative humidity data. Default is "RH".

Value

A tidy data frame containing processed TRH data with columns for Site, Sensor, Date (floored to the nearest hour), Temperature (mean values), and Relative Humidity (mean values).

Examples

```
# Example usage: TRH_data <- tidy_TRHdata("path/to/your/TRHdata.csv")
```

`TRHdata` *Climate dataset to demonstrate functions*

Description

A climate dataset for use to demonstrate how the functions work.

Usage

```
TRHdata
```

Format

A data frame with 35,136 rows and 5 columns:

Site, Sensor Sensor location and name

Date Date is ISOdate time format

Temp, RH Temperature (C) and relative humidity (%) ...

Source

Climate

Index