Package: hydroTools (via r-universe)

May 14, 2026

Title Tools for hydrological model

Version 0.1.8

Description What the package does (one paragraph).

Depends R (>= 4.1)

Imports VIC5 (>= 0.2.4), magrittr, lubridate, dplyr, purrr, data.table, data.tree, DiagrammeR, methods

Suggests testthat (>= 2.1.0), covr, scales, glue, rtop, sf, ggplot2, knitr, rmarkdown

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libuv1-dev libxml2-dev libx11-dev

Repository https://rpkgs.r-universe.dev

Date/Publication 2026-04-14 09:03:38 UTC

RemoteUrl https://github.com/CUG-hydro/hydroTools

RemoteRef HEAD

RemoteSha 34dda9272b56759c83962fddedea5ce08a908b7f

Contents

budyko 2
cal_angle 3
cal_Rl_out 4
cal_Rli 5
cal_Rln 6
cal_Rn 7
cal_Rsi 8

cal_Rsi_toa . . . 9
cal_ssd . . . 10
cal_Ts . . . 10
cal_wetbulb . . . 12
constants . . . 12
convert_temperature . . . 13
correct_basin_net_shapefile . . . 14
detect_groups . . . 15
ET_CR_Xiao2020 . . . 16
ET0_helper . . . 17
ET0_models . . . 19
get_localtime . . . 21
get_sigma . . . 21
heat_index . . . 21
LCL . . . 23
Mw . . . 23
NSE . . . 24
PET_hg . . . 25
plot_calib . . . 26
plot_runoff . . . 26
plot_StreamNet . . . 27
R . . . 27
radiation_Ta . . . 28
soil_class . . . 28
soil_heat_flux . . . 29
SPAW . . . 30
suncalc . . . 31
theta . . . 31
unit_convert . . . 32
usda_sf . . . 33
vapour_press . . . 33
VIC_soilParam . . . 35
XAJ_calib . . . 36
XAJ_predict . . . 37

Index **38**

`budyko` *Budyko Curve*

Description

Budyko Curve

Usage

```
budyko(PET.P, par = 2, method = "Fu1981")

ET_budyko(PET, P, par = 2, method = "Fu1981", ...)

budyko_goal(AET.P, PET.P, par = 2, method = "Fu1981", ...)

budyko_fit(AET.P, PET.P, par = 2, method = "Fu1981", ...)
```

Arguments

`PET.P`	PET divide P, • `PET`: potential evapotranspiration, can be calculated by `ET0_Penman48()`. • `P`: Precipitation • `AET.P`: `AET / P` • `PET.P`: `PET / P`
`par`	scalar value, the parameter of Budyko curve
`method`	one of `c("budyko", "Fu1981", "Zhang2001", "Yang2008", "Pike1964")`, or `all`.
`PET`	potential evapotranspiration (mm)
`P`	precipitation (mm)
`...`	ignored
`AET.P`	`AET / P`

References

1. Budyko, M. I., Climate and Life, 508 pp., Academic, San Diego, Calif., 1974.
2. Fu, Baw-Puh., 1981. On the Calculation of the Evaporation from Land Surface. Chinese Journal of Atmospheric Sciences, 5(1), 23-31.
3. Zhang, L., Dawes, W. R., & Walker, G. R. (2001). Response of mean annual evapotranspiration to vegetation changes at catchment scale. Water Resources Research, 37(3), 701–708. doi:10.1029/2000WR900325
4. Yang, H., Yang, D., Lei, Z., & Sun, F. (2008). New analytical derivation of the mean annual water-energy balance equation. Water Resources Research. doi:10.1029/2007WR006135

`cal_angle` *Rotation angle of two speed vector*

Description

Rotation angle of two speed vector

Usage

```
cal_angle(v1, v2, clockwise = -1)

cal_azimuth(v, refer = c(0, 1))
```

Arguments

v1, v2	speed vector, in deg, (lon, lat). • v1 is the previous time. • v2 is the current time;
clockwise	one of c(-1, 1): -clockwise = -1: clockwise corresponds to negative angle -clockwise = 1: clockwise corresponds to positive angle (azimuth angle in this style)

Value

v2 -> v1:

- rotation angle (in deg, [-180, 180])
  - counterclockwise is positive
  - clockwise is negative
- azimuth angle (in deg, [-180, 180])
  - the angle with north (c(0, 1))

Examples

```
cal_angle(c(1, 0), c(2, 0)) # == 0
cal_angle(c(0, 1), c(0, 2)) # == 0

p1 <- c(1, 0)
p2 <- c(0, 1)

cal_angle(p1, p2) # == 90, counterclockwise
cal_angle(p2, p1) # == -90, clockwise
```

---

cal_Rl_out *black body radiation*

Description

black body radiation

Usage

```
cal_Rl_out(Ta, emiss = 0.97)
```

Arguments

`Ta`	temperature in C deg
`emiss`	Emissivity coefficient

Value

radiation in W m-2

`cal_Rli` *Estimate Incoming longwave radiation.*

Description

Estimate Incoming longwave radiation. Not included in FAO56 paper but added for convinience.

Usage

```
cal_Rli(
  temp,
  ea = NULL,
  s = 1,
  method = c("MAR", "SWI", "IJ", "BRU", "SAT", "KON")
)
```

Arguments

`temp`	Near surface air temperature `[deg Celsius]`.
`ea`	Near surface actual vapour pressure `[kPa]`.
`s`	Cloud air emissivity, the ratio between acutal incomming shortwave radiation and clear sky incoming shortwave radiation.
`method`	The method to esteimate the air emissivity. Must be one of 'MAR', 'SWI', 'IJ', 'BRU', 'T', and 'KON'.

Value

Incomming longwave radiation `[W /m2]`.

References

1. Satterlund, D. R. (1979), An improved equation for estimating long-wave radiation from the atmosphere, Water Resour. Res., 15(6), 1649– 1650, doi:10.1029/WR015i006p01649.
2. Sedlar, J., & Hock, R. (2009). Testing longwave radiation parameterizations under clear and overcast skies at Storglaciären, Sweden. The Cryosphere, 3(1), 75-84.

`cal_Rln` *Net outgoing longwave radiation.*

Description

Net outgoing longwave radiation.

Usage

```
cal_Rln(tmax, tmin, ea, Rsi = NULL, Rsi_o = NULL, cld = NULL)

cal_Rln_yang2019(
  Ts,
  Rsi,
  Rsi_toa,
  lat = 30,
  emiss = 0.96,
  n1 = 2.52,
  n2 = 2.37,
  n3 = 0.035
)
```

Arguments

`tmax`	Daily maximum air temperature at 2m height `[deg Celsius]`.
`tmin`	Daily minimum air temperature at 2m height `[deg Celsius]`.
`ea`	Actual vapor pressure `[kPa]`. Can be estimated by maximum or minimum air temperature and mean relative humidity.
`Rsi`	Surface incoming short-wave radiation (Rsi)
`Rsi_o`	Clear sky incoming shortwave radiation, i. e. extraterrestrial radiation multiply by clear sky transmissivity (i. e. a + b, a and b are coefficients of Angstrom formula. Normally 0.75) `[MJ m-2 day-1]`.
`cld`	Cloud cover `[fraction]`. If provided it would be directly used to calculate net outgoing longwave radiation than Rso.
`Ts`	land surface temperature
`Rsi_toa`	incoming short-wave radiation at the top of the atmosphere
`lat`	latitude (in deg)
`emiss`	Emissivity
`n1`, `n2`, `n3`	parameter for `delta_T`, see Yang 2019 for details

Value

Net outgoing longwave radiation `[MJ m-2 day-1]`

References

1. Yang, Y., & Roderick, M. L. (2019). Radiation, surface temperature and evaporation over wet surfaces. Quarterly Journal of the Royal Meteorological Society, 145(720), 1118–1129. https://doi.org/10.1002/qj.3481
2. Tu, Z., & Yang, Y. (2022). On the Estimation of Potential Evaporation Under Wet and Dry Conditions. Water Resources Research, 58(4). https://doi.org/10.1029/2021wr031486

`cal_Rn` *Surface net radiation*

Description

Surface net radiation

Usage

```
cal_Rn(
  lat,
  J,
  Tmin,
  Tmax,
  ea = NULL,
  RH,
  ssd = NULL,
  cld = NULL,
  Rsi = NULL,
  albedo = 0.23,
  Z = 0,
  ...
)
```

Arguments

`lat`	float, latitude
`J`	integer, day of year
`Tmin`, `Tmax`	numeric vector, min and max 2m-air temperature (degC)
`ea`	Actual vapor pressure (kPa)
`RH`	Relative humidity (%). If `ea` provided, `RH` will be ignored.
`ssd`	numeric vector, sun shine duration (hour)
`cld`	(optional) cloud coverage (0-1). At least one of `cld` and `ssd` should be provided. If `cld` is not null, `ssd` will be ignored.
`Rsi`	(optional) Surface downward shortwave radiation (MJ m-2 d-1). • If not provided, `Rsi` will be calculated by `(as + bs * nN) * Rsi_toa`. • If provided, `ssd` and `cld` will be ignored.

albedo	(optional), Rsn = (1 - albedo) Rsi
Z	(optional) elevation (m), for the calculation of Rsi_o
...	ignored

Value

radiation in [MJ d-1]

- Rn : Surface net radiation
- Rln : Surface outward net longwave radiation (negative means outgoing)
- Rsn : Surface downward net shortwave radiation
- Rsi : Surface downward shortwave radiation (Rsi)
- Rsi_o : Clear-sky surface downward shortwave radiation
- Rsi_toa : Extraterrestrial radiation (top of atmosphere, Rsi_toa) #'

Note

Rn might <= 0. Users need to constrain the min value by pmax(Rn, 0).

Author(s)

Xie YuXuan and Kong Dongdong

See Also

cal_Rsi_toa(), cal_Rsi()

Examples

```
cal_Rn(lat = 30, J = 1, RH = 70, Tmin = 20, Tmax = 30, ssd = 10)
```

---

cal_Rsi *Daily inward shortwave solar radiation.*

Description

Daily inward shortwave solar radiation at crop surface [MJ m-2 day-1] by providing sunshine duration (SSD) in hours or cloud cover in fraction.

Usage

```
cal_Rsi(lat, J, ssd = NULL, cld = NULL, Z = 0, a = 0.25, b = 0.5)
```

Arguments

`lat`	Latitude `[degree]`.
`J`	day of year
`ssd`	sunshine duration lubridate::hours. If `ssd = NULL`, `Rsi` is the clear-sky solar radiation.
`cld`	Cloud cover `[fraction]`. If provided it would be directly used to calculate solar radiation rather than SSD and parameter a and b.
`Z`	elevation (m)
`a`	Coefficient of the Angstrom formula. Determine the relationship between ssd and radiation. Default 0.25.
`b`	Coefficient of the Angstrom formula. Default 0.50.

Value

A data.table, solar radiation at crop surface `[MJ m-2 day-1]`.

- `Rsi_toa`: Clear-sky surface downward shortwave radiation
- `Rsi`: Surface downward shortwave radiation

References

1. Martinezlozano J A, Tena F, Onrubia J E, et al. The historical evolution of the Angstrom formula and its modifications: review and bibliography.data.table::J. Agricultural & Forest Meteorology, 1985, 33(2):109-128.
2. https://github.com/sbegueria/SPEI/blob/master/R/penman.R

See Also

`cal_Rsi_toa()`, `cal_Rn()`

`cal_Rsi_toa` *Daily extraterrestrial radiation (top of atmosphere)*

Description

Estimate daily extraterrestrial radiation `[MJ m-2 day-1]`.

Usage

```
cal_Rsi_toa(lat, J)
```

Arguments

`lat`	Latitude `[degree]`.
`J`	`doy` vector.

Value

extraterrestrial radiation [MJ m-2 day-1].

See Also

cal_Rsi()

Examples

```
cal_Rsi_toa(20, 1)
# [1] 25.84874
```

---

cal_ssd *sunset hour angle*

Description

Calculating sunset hour angle (ws) according to Allen Eq. 25.

Usage

```
cal_sunset_angle(lat, J)

ws2ssd(ws)

cal_ssd(lat, J)
```

Arguments

lat	Latitude [degree].
J	doy vector.
ws	sunset hour angle

---

cal_Ts *Evaporative surface temperature*

Description

Evaporative surface temperature (land surface, Tland; or leaf surface Tleaf). Land surface temperature infered by Monteith 1965 Equation.

Usage

```
cal_Ts(
  Rn,
  Tair,
  D,
  U2,
  Pa = atm,
  rH = NULL,
  rs = 0,
  method = c("simple", "full", "ma2021"),
  ...
)
```

Arguments

`Rn`	land surface net radiation, W m-2
`Tair`	2m air temperature (degC)
`D`	vapor pressure deficit (kPa)
`U2`	wind speed at 2m
`Pa`	surface air pressure (kPa)
`rH`	conductance for heat
`rs`	If `rs = 0`, Monteith 1965 leaf evaporation Equation becomes Penman 1948 water evaporation. ignore the influence of Ts on net cal_radiation
`method`	• `simple`: Monteith 1965 Equation • `full` (not finished): Yang 2019 • `ma2021`: Ma 2021
`...`	ignored

Details

- `rH`: aerodynamic resistance of heat
- `rs`: stamotal resistance of water

References

1. Monteith, J. P. (1965). An introduction to environmental physics.
2. Yang, Y., & Roderick, M. L. (2019). Radiation, surface temperature and evaporation over wet surfaces. Quarterly Journal of the Royal Meteorological Society, 145(720), 1118–1129. https://doi.org/10.1002/qj.3481
3. Ma, N., Szilagyi, J., & Zhang, Y. (2021). Calibration-Free Complementary Relationship Estimates Terrestrial Evapotranspiration Globally. Water Resources Research, 57(9), 1–27. https://doi.org/10.1029/2021WR029691

Examples

```
library(dplyr)
# The relationship of Rn, Ts
Rn = 0:200
cal_Ts(Rn, 25, D = 1, U2 = 2)
cal_Ts(200, 25, D = 1, U2 = 2)
# plot(Rn, dat$Ts, type = "l", main = "(a) Ts ~ Rn")
# plot(Rn, dat$Eeq, type = "l", main = "(b) Eeq ~ Rn")
# # plot(Rn, dat$Evp, type = "l") # a constant value
# dat %<>% mutate(Rn = Rn, bowen = ET0 / (Rn * 0.086400 / lambda - ET0))
# plot(bowen ~ Rn, dat, type = "l", main = "(b) Eeq ~ Rn")
```

---

cal_wetbulb *wetbulb temperature*

Description

wetbulb temperature

Usage

```
cal_wetbulb(Tair, Tdew, Pa = atm)

cal_Tw(ea, Tair, Pa = atm)
```

Arguments

Tair	2m air temperature (degC)
Pa	surface air pressure (kPa)
ea	Actual vapor pressure (kPa)

References

1. https://github.com/wrf-model/WRF/blob/master/phys/module_diag_functions.F#L1154

---

constants *constants in meteorology*

Description

- T0: Absolute zero in Kelvin T_0 (K)
- Es.T0: $e_s(T_0) = 6.11hPa$ is the saturation vapor pressure at the absolute zero $T_0 = 273.15K$.
- L: Latent heat of water vapor $L = 2.5 \times 10^6 J/kg$
- atm: standard pressure at sea surface, `101.325 kPa`
- Cp: specific heat at constant pressure, `1.013 * 1e-3 MJ kg-1 degC-1`

Usage

```
T0

K0

L

Es.T0

atm
```

Format

An object of class `numeric` of length 1.

An object of class `numeric` of length 1.

An object of class `numeric` of length 1.

An object of class `numeric` of length 1.

An object of class `numeric` of length 1.

---

convert_temperature *Convert from one temperature metric to another*

Description

This function allows you to convert a vector of temperature values between Fahrenheit, Celsius, and degrees Kelvin.

Usage

```
convert_temperature(temperature, old_metric, new_metric)

celsius.to.fahrenheit(T.celsius)

celsius.to.kelvin(T.celsius)

fahrenheit.to.celsius(T.fahrenheit)

fahrenheit.to.kelvin(T.fahrenheit)

kelvin.to.celsius(T.kelvin)

kelvin.to.fahrenheit(T.kelvin)
```

Arguments

`temperature`	A numeric vector of temperatures to be converted.
`old_metric`	The metric from which you want to convert. Possible options are: • `fahrenheit`, `f` • `kelvin`, `k` • `celsius`, `c`
`new_metric`	The metric to which you want to convert. The same options are possible as for `old_metric`.
`T.kelvin`, `T.fahrenheit`, `T.celsius`	Numeric vector of temperatures in Kelvin, Fahrenheit and Celsius.

Value

A numeric vector with temperature converted to the metric specified by the argument `new_metric`.

A numeric vector of temperature values in the new unit.

Author(s)

Joshua Ferreri `<joshua.m.ferreri@gmail.com>`, Brooke Anderson `<brooke.anderson@colostate.edu>`, Roger Peng `<rdpeng@gmail.com>`

References

1. online heat index calculator
2. online temperature converter

`correct_basin_net_shapefile`
correct_basin_net_shapefile

Description

correct_basin_net_shapefile

Usage

```
correct_basin_net_shapefile(
  ID,
  shp,
  tree,
  outdir = "basins/subbasin",
  prefix = "shed_ ",
  overwrite = FALSE
)
```

Arguments

`ID`	the ID of basins
`shp`	basin shapefile, sf polygon object
`tree`	returned by `plot_StreamNet()`
`outdir`	outdir of shapefile
`prefix`	prefix of shapefile

`detect_groups` *detect_flood_events*

Description

detect_flood_events

flood_divide

Usage

```
detect_groups(df, inds, gap_max = 5, extend = ddays(5))

detect_flood_events(
  date,
  Q,
  Q_min = 2,
  Q_peak = 10,
  gap_max = 5,
  extend = ddays(5)
)

flood_divide(df, ...)

merge_flood(df, info_flood, format = "%Y.%m")
```

Arguments

`df`	A data.table with date and Q columns
`gap_max`	Default `5` days. • `gaps` (in days) <= `gap_max` is regarded as the same event. • `gaps` > `gap_max`, it will be regarded as two events.
`extend`	Default `ddays(5)`. Extend `nday` in the left and right of a event
`Q_min`	minimum discharge to detect flood events
`Q_peak`	peak discharge to detect flood events
`...`	parameters passed to `detect_flood_events()`

`ET_CR_Xiao2020` *ET complementary*

Description

ET complementary

Usage

```
ET_CR_Xiao2020(
  Rn,
  Tair,
  D,
  U2,
  Pa = atm,
  method = c("simple", "full", "ma2021"),
  rs = 0
)

ET_CR_Ma2021(
  Rn,
  Tair,
  D,
  U2,
  Pa = atm,
  method = c("simple", "full", "ma2021"),
  rs = 0
)
```

Arguments

`Rn`	land surface net radiation, W m-2
`Tair`	2m air temperature (degC)
`D`	vapor pressure deficit (kPa)
`U2`	2m wind speed (m/s)
`Pa`	surface air pressure (kPa)
`method`	method for `Tw` calculation (not used in `ET_CR_Xiao2020`), see reference for details
`rs`	stamotal conductance for water on capony scale

References

1. Xiao, M., Yu, Z., Kong, D., Gu, X., Mammarella, I., Montagnani, L., … Gioli, B. (2020). Stomatal response to decreased relative humidity constrains the acceleration of terrestrial evapotranspiration. Environmental Research Letters, 15(9). doi:10.1088/1748-9326/ab9967

2. Ma, N., Szilagyi, J., & Zhang, Y. (2021). Calibration-Free Complementary Relationship Estimates Terrestrial Evapotranspiration Globally. Water Resources Research, 57(9), 1–27. https://doi.org/10.1029/2021WR029691

Examples

```
ET_CR_Xiao2020(250, 25, D = 1, U2 = 2)
ET_CR_Ma2021(250, 25, D = 1, U2 = 2)
```

`ET0_helper` *helper functions for potential evapotranspiration (ET0)*

Description

- `lambda`: latent heat of vaporization, about `[2.5 MJ kg-1]`.
- `slope`: The slope of the saturation vapour pressure curve at certain air temperature `Tair`, `[kPa degC-1]`.
- `gamma`: psychrometric constant (`[kPa degC-1]`), `Cp*Pa/(epsilon*lambda)`.
- `U2`: 10m wind speed (m/s). According to wind profile relationship, convert U_z to U_2.
- `es`: saturation vapor pressure (kPa)
- `ea`: actual vapor pressure (kPa)
  1. `RH_mean` | `Tmin`, `Tmax`: `(es(Tmax) + es(Tmin)) * RH_mean/200`
  2. `Tmin` : `es(Tmin)`
- `cal_TvK`: vitual temperature (K)
  1. Tair + q
  2. Tair + ea/Pa
  3. `1.01 * (Tair + 273)` , FAO56 Eq. 3-7
- `cal_rho_a`: air density (kg/m^3)
- `rH`: resistance of aerodynamic (s/m), about `100 s/m`.

Usage

```
cal_U2(Uz, z.wind = 10)

cal_ea(Tmin, Tmax = NULL, RH_mean = NULL)

cal_VPD(Tair, Tdew = NULL)

cal_lambda(Tair)

cal_slope(Tair)

cal_gamma(Tair, Pa = atm)
```

```
cal_bowen(Tair, Pa = atm)

cal_Pa(z = NULL)

cal_rH(U2, h = 0.12)

cal_rH2(U2, Tair, Pa = atm)

cal_TvK(Tair, q = NULL, ea = NULL, Pa = atm)

cal_rho_a(Tair, Pa = atm, q = NULL, ea = NULL)

cal_es(Tair)
```

Arguments

`Uz`	wind speed at height `z.wind`
`z.wind`	Height where measure the wind speed `[m]`. Default 10m.
`Tmin`	Daily minimum air temperature at 2m height `[deg Celsius]`
`Tmax`	Daily maximum air temperature at 2m height `[deg Celsius]`
`RH_mean`	daily mean relative humidity `[%]`
`Tair`	2m air temperature, (`[deg Celsius]`)
`Tdew`	dew temperature, (`[deg Celsius]`)
`Pa`	land surface Air pressure `[kPa]`.
`z`	Elevation above sea level `[m]`. Must provided if pres are not provided.
`U2`	wind speed at 2m
`h`	canopy height (m), for the calculation of the following intermediate variables: • `d`: zero plane displacement height (m) • `z_om`: roughness length governing momentum transfer (m) • `z_oh`: roughness length governing transfer of heat and vapour (m) • `z_h` : height of humidity measurements (m) • `z_m` : height of wind measurements (m)
`q`	specific humidity (g/g)
`ea`	actual vapor pressure (kPa)

References

1. Allen, R. G., & Luis S. Pereira. (1998). Crop evapotranspiration-Guidelines for computing crop water requirements-FAO Irrigation and drainage paper 56. European Journal of Agronomy, 34(3), 144–152. doi:10.1016/j.eja.2010.12.001

Examples

```
cal_VPD(10, 5)

par(mfrow = c(2, 2), mar = c(3, 1.8, 2, 1), mgp = c(2, 0.6, 0))
Tair <- -10:50
plot(cal_es(-10:50), main = "es (kPa)", xlab = "Tair")
plot(cal_bowen(-10:50), main = "Bowen ratio", xlab = "Tair")
plot(cal_slope(-10:50), main = "slope (kPa/degC)", xlab = "Tair")
plot(cal_gamma(-10:50), main = "gamma (kPa/degC)", xlab = "Tair")
```

ET0_models *Potential Evapotranspiration models*

Description

- `ET0_eq` : Equilibrium evaporation, `slope / (slope + gamma) * Rn`
- `ET0_Penman48` : Penman 1948 equation simplified by Shuttleworth 1993
- `ET0_Monteith65`: Penman-Monteith 1965
- `ET0_PT72` : Priestley Taylor 1972, `ET0_PT72 = ET0_eq * 1.26`
- `ET0_FAO98` : Penman-Monteith reference crop evapotranspiration, FAO56

Usage

```
ET0_eq(Rn, Tair, Pa = atm, ...)

ET0_Penman48(Rn, Tair, Pa = atm, D, wind, z.wind = 2)

ET0_Monteith65(
  Rn,
  Tair,
  Pa = atm,
  D,
  wind,
  z.wind = 2,
  rs = 70,
  rH = NULL,
  ...
)

ET0_PT72(Rn, Tair, Pa = atm, alpha = 1.26, ...)

ET0_FAO98(Rn, Tair, Pa = atm, D, wind, z.wind = 2, tall_crop = FALSE)
```

Arguments

`Rn`	land surface net radiation, W m-2
`Tair`	2m air temperature (degC)
`Pa`	surface air pressure (kPa)
`D`	vapor pressure deficit (kPa)
`wind`	wind speed at the height of `z.wind`
`z.wind`	Height where measure the wind speed `[m]`. Default 2m.
`alpha`	albedo `[0-1]`
`tall_crop`	tall crop or not, see FAO1998 for details

Value

- `lambda`: Latent heat of vaporization, about `[2.5 MJ kg-1]`.
- `slope` : The slope of the saturation vapour pressure curve at certain air temperature Tair, `[kPa degC-1]`.
- `gamma` : Psychrometric constant (`[kPa degC-1]`), `Cp*Pa/(epsilon*lambda)`
- `Eeq` : Equilibrium (radiative) evaporation, mm
- `Evp` : Aerodynamic evaporation, mm
- `ET0` : Potential evapotranspiration, mm

References

1. Penman equation. Wikipedia 2021. https://en.wikipedia.org/w/index.php?title=Penman_equation
2. Shuttleworth, W. J. (1993) Evaporation. In: Handbook of Hydrology (ed. by D. Maidment). McGraw-Hill, New York.
3. Allen, R. G., & Luis S. Pereira. (1998). Crop evapotranspiration-Guidelines for computing crop water requirements-FAO Irrigation and drainage paper 56. European Journal of Agronomy, 34(3), 144-152. doi:10.1016/j.eja.2010.12.001
4. Allen, R. G., Pereira, L. S., Raes, D., & Smith, M. (1998). FAO Irrigation and drainage paper No. 56. Rome: Food and Agriculture Organization of the United Nations, 56(97), e156.
5. Chapter 2 - FAO Penman-Monteith equation, https://www.fao.org/3/x0490E/x0490e06.htm

Examples

```
ET0_PT72(250, 25, D = 1)
ET0_Penman48(250, 25, D = 1, wind = 2)
ET0_FAO98(250, 25, D = 1, wind = 2)
ET0_Monteith65(250, 25, D = 1, wind = 2)
```

get_localtime *Get local time*

Description

Get local time

Usage

```
get_localtime(time, lon, timeZone = 8)
```

Arguments

time	POSIXct object
lon	longitude in deg

get_sigma *Sun Declination angle*

Description

Sun Declination angle

Usage

```
get_sigma(J)
```

heat_index *heat index (Human-perceived temperature)*

Description

`heat_index` converts a numeric scalar of temperature (in Fahrenheit) and a numeric scalar of relative humidity (in %) to heat index (in Fahrenheit). This function is not meant to be used outside of the `heat_index_vec()` function.

Usage

```
heat_index(t, rh)

.heat.index(t = NA, rh = NA)

heat_index_vec(t = NA, rh = NA)
```

Arguments

t	Numeric scalar of air temperature, in celsius.
rh	Numeric scalar of relative humidity, in %.

Details

If an impossible value of relative humidity is given (below 0% or above 100%), heat index is returned as NA.

Value

A numeric scalar of heat index, in celsius

Note

Equations are from the source code for the US National Weather Service's online heat index calculator.

Author(s)

Brooke Anderson <brooke.anderson@colostate.edu>, Roger Peng <rdpeng@gmail.com>

References

1. Anderson GB, Bell ML, Peng RD. 2013. Methods to calculate the heat index as an exposure Metric in environmental health research. Environmental Health Perspectives 121(10):1111-1119.
2. National Weather Service Hydrometeorological Prediction Center Web Team. Heat Index Calculator. 30 Jan 2015. http://www.wpc.ncep.noaa.gov/html/heatindex.shtml. Accessed 18 Dec 2015.
3. Rothfusz L. 1990. The heat index (or, more than you ever wanted to know about heat index) (Technical Attachment SR 90-23). Fort Worth: Scientific Services Division, National Weather Service.
4. R. Steadman, 1979. The assessment of sultriness. Part I: A temperature-humidity index based on human physiology and clothing science. Journal of Applied Meteorology, 18(7):861–873.

Examples

```
n <- 1e3
t <- seq(-50, 100, length.out = n)
rh <- seq(0, 100, length.out = length(t))

## bug exist
i <- 88
heat_index(t[i], rh[i])
heat_index_vec(t[i], rh[i])
# heat_index_julia(t[i], rh[i])
```

```
# julia_setup()
r <- heat_index(t, rh)
r_vec <- heat_index_vec(t, rh)
# r_jl <- heat_index_julia(t, rh)

# add tests for julia NA values
# r1 <- heat_index_julia(t*NA, rh*NA)

all.equal(r, r_vec)
# all.equal(r_jl, r_vec)

# microbenchmark::microbenchmark(
#     r2 <- heat_index(t, rh),
#     r1 <- heat_index_julia(t, rh)
#     # r_vec <- heat_index_vec(t, rh)
# )
```

`LCL` *Lifted Condensation Level (LCL)*

Description

- `LCL` : Bolton 1980, Eq. 15
- `LCL_RH`: Bolton 1980, Eq. 22

Usage

```
LCL(P0, T0, Td)

LCL_RH(T, RH)
```

Arguments

`P0`	pressure at surface (hPa)
`T0`	temperature at surface (C)
`Td`	dew point temperature (C)

`Mw` *Molecular weight*

Description

`Mw`: Molecular weight of dry air $M_d = 28.9634 g/mol$ `Md`: Molecular weight of water vapor $M_w = 18.01528 g/mol$ `epsilon`: the ratio of `Mw` to `Md`

Usage

```
Mw
```

```
Md
```

```
epsilon
```

Format

An object of class `numeric` of length 1.

An object of class `numeric` of length 1.

An object of class `numeric` of length 1.

`NSE` *GOF*

Description

Good of fitting

Usage

```
NSE(yobs, ysim, w, ...)
```

```
KGE(yobs, ysim, ...)
```

```
GOF(yobs, ysim, w, include.cv = FALSE, include.r = TRUE)
```

Arguments

`yobs`	Numeric vector, observations
`ysim`	Numeric vector, corresponding simulated values
`w`	Numeric vector, weights of every points. If w included, when calculating mean, Bias, MAE, RMSE and NSE, w will be taken into considered.
`...`	ignored
`include.cv`	If true, cv will be included.
`include.r`	If true, r and R2 will be included.

Value

- `RMSE` root mean square error
- `NSE` NASH coefficient
- `MAE` mean absolute error
- `AI` Agreement index (only good points (w == 1)) participate to calculate. See details in Zhang et al., (2015).

- `Bias` bias
- `Bias_perc` bias percentage
- `n_sim` number of valid obs
- `cv` Coefficient of variation
- `R2` correlation of determination
- `R` pearson correlation
- `pvalue` pvalue of `R`

References

1. https://en.wikipedia.org/wiki/Coefficient_of_determination
2. https://en.wikipedia.org/wiki/Explained_sum_of_squares
3. https://en.wikipedia.org/wiki/Nash%E2%80%93Sutcliffe_model_efficiency_coefficient
4. Zhang Xiaoyang (2015), http://dx.doi.org/10.1016/j.rse.2014.10.012

Examples

```
yobs = rnorm(100)
ysim = yobs + rnorm(100)/4
GOF(yobs, ysim)
```

`PET_hg` *Estimate ET0 by Hargreaves equation.*

Description

Estimate reference evapotranspiration (ET0) from a hypothetical short grass reference surface using the the Hargreaves equation.

Usage

```
PET_hg(Tmax, Tmin, Tavg = NULL, Rsi_toa = NULL, lat = NULL, dates = NULL)
```

Arguments

`Tmax`	Daily maximum air temperature at 2m height `[deg Celsius]`.
`Tmin`	Daily minimum air temperature at 2m height `[deg Celsius]`.
`Tavg`	Daily mean air temperature at 2m height `[deg Celsius]`. If not provided it would estimated by averaging the Tmax and tmin.
`Rsi_toa`	Clear sky incoming shortwave radiation, i. e. extraterrestrial radiation multiply by clear sky transmissivity (i. e. a + b, a and b are coefficients of Angstrom formula. Normally 0.75) `[MJ m-2 day-1]`. If not provided, must provide lat and dates.
`lat`	Latitude `[degree]`.
`dates`	A R Date type of a vector of Date type. If not provided, it will Regard the ssd series is begin on the first day of a year.

Value

Reference evapotranspiration ET0 from a hypothetical grass reference surface `[mm day-1]`.

`plot_calib` *plot_calib*

Description

plot_calib

Usage

```
plot_calib(data, ...)
```

Arguments

`data`	with the columns of `date`, `prcp`, `Qsim`, `Qobs`
`...`	others to plot precipitation

`plot_runoff` *plot_runoff*

Description

plot_runoff

Usage

```
plot_runoff(
  df_q,
  df_prcp = NULL,
  xlim = NULL,
  ylim2 = c(50, 0),
  legend.position = c(1, 1),
  legend.justification = c(1, 1),
  ...
)
```

Arguments

`df_q`	A data.frame with the columns of `date`, `Qsim`, `Qobs`. At least one of `Qsim` and `Qobs` should be provided.
`df_prcp`	A data.frame with the columns of `date`, `prcp`
`xlim`	limit of x axis
`ylim2`	limit of second y axis (for precipitation)
`...`	other parameters passed to `plot()` for precipitation plot

plot_StreamNet *plot_tree*

Description

plot_tree

Usage

```
plot_StreamNet(info, title = "", fout = "tree.pdf", show = FALSE, root = -1)
```

Arguments

info	A data.table with the column of ["id", "iddown", "name"]
title	The title of the tree

R *Specific gas constants*

Description

- R: gas constant, J/(mol K)
- Rw: Specific gas constant of water vapor $R_w = \frac{1000R}{M_w} = 461.52J/(kgK)$.
- Rd: Specific gas constant of dry air (J/(kg K)).

Usage

```
R

Rw

Rd

Cp
```

Format

An object of class numeric of length 1.

An object of class numeric of length 1.

An object of class numeric of length 1.

An object of class numeric of length 1.

radiation_Ta *air temperature under solar radiation*

Description

air temperature under solar radiation

Usage

```
radiation_Ta(Rs, Rl = 0, albedo = 0.3, emiss = 0.97)

cal_radiation_Ta(
  Ta,
  Rs,
  Rl = 0,
  dt = 3600,
  method = c("approx", "exact"),
  ...,
  albedo = 0.3,
  emiss = 0.97
)
```

Arguments

Rs, Rl	numeric vector, inward shortwave and longwave radiation, in W/m2
Ta	scalar, the initial air temperature, in degC
dt	delta t, in second

soil_class *soil texture class, based on USDA triangle*

Description

soil texture class, based on USDA triangle

Usage

```
soil_class(clay, sand)
```

Arguments

clay	Vector, percent of clay
sand	Vector, percent of sand

Value

Character vector of soil texture class, one of `c("Cl", "SiCl", "SaCl", "ClLo", "SiClLo", "SaClLo", "Lo", "SiLo", "SaLo", "Si", "LoSa", "Sa")`.

References

1. Julien Moeys (2016). soiltexture: Functions for Soil Texture Plot, Classification and Transformation. R package version 1.5.1. https://CRAN.R-project.org/package=soiltexture

Examples

```
clay = c(05, 60, 15, 05, 25, 05, 25, 45, 65, 75, 13, 47)
sand = c(90, 32, 70, 70, 20, 10, 10, 10, 20, 10, 70, 10)
soil_class(clay, sand)
# c("Sa", "Cl", "SaLo", "SaLo", "SiLo", "Si", "SiLo", "SiCl", "Cl", "Cl", "SaLo", "SiCl")
```

`soil_heat_flux` *Soil heat flux.*

Description

Estimate monthly soil heat flux (G) from the mean air temperature of the previous, current or next month assuming as grass crop.

Usage

```
soil_heat_flux(t.p, t.n = NULL, t.c = NULL)
```

Arguments

`t.p`	Mean air temperature of the previous month `[Celsius]`.
`t.n`	Mean air temperature of the next month `[Celsius]`.
`t.c`	Mean air temperature of the current month `[Celsius]`.

Value

Soil heat flux `[MJ m-2 day-1]`.

SPAW *Soil properties in SPAW*

Description

Wilting point, field capacity and saturated moisture

- `wilting point`: 1500 kPa moisture, %v, (Eq. 1)
- `field capacity`: 33 kPa moisture, %v, (Eq. 2)
- `saturated moisture`: Saturation (0 kPa moisture), %v, (Eq. 5)

Usage

```
wilting_point(S, C, OM)

field_capacity(S, C, OM)

saturated_mois(S, C, OM)

wilting_point_salinity(S = 0.2, C = 0.2, OM = 2.5, EC = 3)
```

Arguments

`S`	the weight ratio of Sand, (weight ratio, 0-1)
`C`	the weight ratio of Clay, (weight ratio, 0-1)
`OM`	the percentage of Organic Matter, (%w, 0-100)
`EC`	Electrical conductance of a saturated soil extract (dS/m = mili-mho/cm)

Value

%v

Note

The unit of `OM` is `%w`, which is different from `S` and `C`.

References

1. Saxton, K. E., & Rawls, W. J. (2006). Soil water characteristic estimates by texture and organic matter for hydrologic solutions. Soil Science Society of America Journal, 70(5), 1569-1578. doi:10.2136/sssaj2005.0117

Examples

```
S = C = 0.20; OM = 2.5
wilting_point(S, C, OM)  # 13.8%
field_capacity(S, C, OM) # 32.1%
saturated_mois(S, C, OM) # 48.2%
```

`suncalc` *suncalc*

Description

The default location is ZuoLing.

Usage

```
suncalc(
  time,
  lon = 114.6053,
  lat = 30.49694,
  ...,
  year = year(Sys.time()),
  verbose = TRUE
)

sunrise(date = Sys.Date(), lon = 80, lat = 30, timeZone = 8)
```

Examples

```
suncalc(Sys.Date())

sunrise(Sys.Date())
```

`theta` *equivalent potential temperature*

Description

equivalent potential temperature

Usage

```
theta(P0, T0)

theta_wet(P0, T0, Td)

theta_wet_bolton(P0, T0, Td)
```

Arguments

`P0`	pressure at surface (hPa)
`T0`	temperature at surface (Cdeg)
`Td`	dew point temperature (Cdeg)

References

1. https://unidata.github.io/MetPy/latest/api/generated/metpy.calc.equivalent_potential_temperature.html
2. https://github.com/wrf-model/WRF/blob/master/phys/module_diag_functions.F#L122
3. https://github.com/Unidata/MetPy/blob/e0e24d51702787943fc3c0481fa9a6632abe9d20/src/metpy/calc/the

Examples

```
theta_wet(850, 20, 18)
theta_wet_bolton(850, 20, 18)
```

---

unit_convert *unit convertion*

Description

- `R2Q`: Q(m^3/s) = R(mm) * area (km^2) * 1000/(3600*24)
- `W2mm`: 1 MJ /m2/day =0.408 mm /day, 1 Watt /m2 = 0.0864 MJ /m2/day

Usage

```
R2Q(R, area = dt * 3.6, dt = 24)

Q2R(Q, area = dt * 3.6, dt = 24)

MJ_2W(x)

MJ_2mm(x)

W2_MJ(x)

W2mm(x, tmean = 0)

K2T(x)

T2K(x)

deg2rad(deg)

rad2deg(rad)
```

Arguments

`R`	runoff (mm per dt)
`area`	basin area (km^2), if area = ,
`dt`	time duration (hour)

`Q`	runoff flow (m^3/s)
`x`	scalar or numeric vector
`tmean`	daily mean temperature
`deg`	angle in degree
`rad`	angle in radian

References

http://www.fao.org/3/X0490E/x0490e0i.htm

`usda_sf` *USDA soil class classification*

Description

USDA soil class classification

Usage

```
usda_sf
```

Format

An object of class `sf` (inherits from `data.frame`) with 12 rows and 2 columns.

References

1. https://en.wikipedia.org/wiki/Soil_texture

`vapour_press` *Helper functions for vapour pressure*

Description

Helper functions for vapour pressure

Usage

```
w2q(w)

q2w(q, Pa = atm)

q2ea(q, Pa = atm)

w2ea(w, Pa = atm)

ea2VPD(ea, RH)

vapour_press(q, Pa = atm)

q2RH(q, Tair, Pa = atm)

RH2q(RH, Tair, Pa = atm)

q_from_RH(RH, Tair, Pa = atm)

cal_qs(Tair, Pa = atm)

cal_ws(Tair, Pa = atm)

ea2w(ea, Pa = atm)

ea2q(ea, Pa = atm)

RH2q(RH, Tair, Pa = atm)

Tdew2q(Tdew, Pa = atm)

Tdew2w(Tdew, Pa = atm)

Tdew2RH(Tdew, Tair)

Tdew_from_q(q, Pa = atm)

Tdew_from_w(w, Pa = atm)
```

Arguments

w	mix ratio, m_w / m_d
q	specific humidity in kg/kg or g/g
Pa	surface air pressure
ea	actual vapor pressure (kPa)
RH	relative humidity, in %
Tair	air temperature, in degC

`Tdew` dew temperature (in degC)

Value

e in the same unit as Pa

References

1. https://www.eol.ucar.edu/projects/ceop/dm/documents/refdata_report/eqns.html, Eq-17
2. Bolton, David. "The Computation of Equivalent Potential Temperature." Monthly Weather Review 108, no. 7 (July 1980): 1046–53. <https://doi.org/10.1175/1520-0493(1980)108<1046:TCOEPT>2.0.CO;2>.

https://earthscience.stackexchange.com/questions/2360/how-do-i-convert-specific-humidity-to-relative-humidity

Examples

```
RH = 90
Pa = atm
Tair = 30
q = RH2q(RH, Pa, Tair)
RH2 = q2RH(q, Pa, Tair)
e = q2ea(q, Pa)
es = cal_es(Tair)
```

`VIC_soilParam` *VIC model soil parameters*

Description

Soil parameters for VIC model (calculated from HWSD database)

- `expt` : 3+2b (Eq. 17)
- `Kstat`: Saturated conductivity (mm/hr) (Eq. 16)
- `bubble`: Bubbling pressure (cm) (Eq. 4)
- `Wcr_FT` : Field Capacity, Fractional soil moisture content at the critical point (~70% of field capacity) (fraction of maximum moisture)
- `Wpwp_FT` : Wilting Point, Fractional soil moisture content at the wilting point (fraction of maximum moisture)

Usage

```
expt(S, C, OM)

Ksat(S, C, OM)

bubble(S, C, OM)

Wcr_FT(S, C, OM)

Wpwp_FT(S, C, OM)
```

Arguments

S	the weight ratio of Sand, (weight ratio, 0-1)
C	the weight ratio of Clay, (weight ratio, 0-1)
OM	the percentage of Organic Matter, (%w, 0-100)

Details

- Kstat: unit has changed from `mm/h` to `mm/day`

References

Saxton, K. E., & Rawls, W. J. (2006). Soil water characteristic estimates by texture and organic matter for hydrologic solutions. Soil Science Society of America Journal, 70(5), 1569-1578. https://doi.org/10.2136/sssaj2005.0117

Examples

```
S = C = 0.20; OM = 2.5
expt(S, C, OM)
Ksat(S, C, OM)           # 12.19
bubble(S, C, OM)
field_capacity(S, C, OM) # 32.1%
wilting_point(S, C, OM)  # 13.8%
# Fractional soil moisture
Wcr_FT(S, C, OM)
Wpwp_FT(S, C, OM)
```

XAJ_calib *XAJ model Parameter calibration*

Description

XAJ model Parameter calibration

Usage

```
XAJ_calib(
  Qobs,
  prcp,
  ET0,
  area,
  dt = 24,
  date = NULL,
  maxn = 1000,
  index = "KGE",
  seed = 1,
  ...
)
```

Arguments

Qobs	Observed Total runoff, (m^3/s)
prcp	Precipitation (mm/d)
ET0	Pan evaporation or potential evapotranspiration (mm/d)
area	basin area (km^2).
dt	time step (hour)
date	(optional) corresponding date of Qobs
index	KGE or NSE
seed	(can be ignored) starting number of random number generator, see base::set.seed() for details. This parameter is to make sure optimization result is same in different tries.
...	ignored

XAJ_predict *XAJ_predict*

Description

XAJ_predict

Usage

```
XAJ_predict(model, newdata)
```

Arguments

model	XAJ model returned by XAJ_calib()
newdata	A data.frame, with the column of prcp, ET0, date (optional).

Index

∗ **datasets**
    constants, 12
    Mw, 23
    R, 27
    usda_sf, 33
.heat.index *(heat_index)*, 21

atm *(constants)*, 12

base::set.seed(), *37*
bubble *(VIC_soilParam)*, 35
budyko, 2
budyko_fit *(budyko)*, 2
budyko_goal *(budyko)*, 2

cal_angle, 3
cal_azimuth *(cal_angle)*, 3
cal_bowen *(ET0_helper)*, 17
cal_ea *(ET0_helper)*, 17
cal_es *(ET0_helper)*, 17
cal_gamma *(ET0_helper)*, 17
cal_lambda *(ET0_helper)*, 17
cal_Pa *(ET0_helper)*, 17
cal_qs *(vapour_press)*, 33
cal_radiation_Ta *(radiation_Ta)*, 28
cal_rH *(ET0_helper)*, 17
cal_rH2 *(ET0_helper)*, 17
cal_rho_a *(ET0_helper)*, 17
cal_Rl_out, 4
cal_Rli, 5
cal_Rln, 6
cal_Rln_yang2019 *(cal_Rln)*, 6
cal_Rn, 7
cal_Rn(), *9*
cal_Rsi, 8
cal_Rsi(), *8*, *10*
cal_Rsi_toa, 9
cal_Rsi_toa(), *8*, *9*
cal_slope *(ET0_helper)*, 17
cal_ssd, 10
cal_sunset_angle *(cal_ssd)*, 10
cal_Ts, 10
cal_TvK *(ET0_helper)*, 17
cal_Tw *(cal_wetbulb)*, 12
cal_U2 *(ET0_helper)*, 17
cal_VPD *(ET0_helper)*, 17
cal_wetbulb, 12
cal_ws *(vapour_press)*, 33
celsius.to.fahrenheit
        (convert_temperature), 13
celsius.to.kelvin
        (convert_temperature), 13
constants, 12
convert_temperature, 13
correct_basin_net_shapefile, 14
Cp *(R)*, 27

data.table::J, *9*
deg2rad *(unit_convert)*, 32
detect_flood_events *(detect_groups)*,
        15
detect_flood_events(), *15*
detect_groups, 15

ea2q *(vapour_press)*, 33
ea2VPD *(vapour_press)*, 33
ea2w *(vapour_press)*, 33
epsilon *(Mw)*, 23
Es.T0 *(constants)*, 12
ET0_eq *(ET0_models)*, 19
ET0_FAO98 *(ET0_models)*, 19
ET0_helper, 17
ET0_models, 19
ET0_Monteith65 *(ET0_models)*, 19
ET0_Penman48 *(ET0_models)*, 19
ET0_Penman48(), *3*
ET0_PT72 *(ET0_models)*, 19
ET_budyko *(budyko)*, 2
ET_CR_Ma2021 *(ET_CR_Xiao2020)*, 16
ET_CR_Xiao2020, 16

expt *(VIC_soilParam)*, 35

fahrenheit.to.celsius
(convert_temperature), 13
fahrenheit.to.kelvin
(convert_temperature), 13
field_capacity *(SPAW)*, 30
flood_divide *(detect_groups)*, 15

get_localtime, 21
get_sigma, 21
GOF *(NSE)*, 24

heat_index, 21
heat_index_vec *(heat_index)*, 21
heat_index_vec(), *21*

K0 *(constants)*, 12
K2T *(unit_convert)*, 32
kelvin.to.celsius
(convert_temperature), 13
kelvin.to.fahrenheit
(convert_temperature), 13
KGE *(NSE)*, 24
Ksat *(VIC_soilParam)*, 35

L *(constants)*, 12
LCL, 23
LCL_RH *(LCL)*, 23
lubridate::hours, *9*

Md *(Mw)*, 23
merge_flood *(detect_groups)*, 15
MJ_2mm *(unit_convert)*, 32
MJ_2W *(unit_convert)*, 32
Mw, 23

NSE, 24

PET_hg, 25
plot(), *26*
plot_calib, 26
plot_runoff, 26
plot_StreamNet, 27
plot_StreamNet(), *15*

q2ea *(vapour_press)*, 33
Q2R *(unit_convert)*, 32
q2RH *(vapour_press)*, 33
q2w *(vapour_press)*, 33
q_from_RH *(vapour_press)*, 33

R, 27
R2Q *(unit_convert)*, 32
rad2deg *(unit_convert)*, 32
radiation_Ta, 28
Rd *(R)*, 27
RH2q *(vapour_press)*, 33
Rw *(R)*, 27

saturated_mois *(SPAW)*, 30
soil_class, 28
soil_heat_flux, 29
SPAW, 30
suncalc, 31
sunrise *(suncalc)*, 31

T0 *(constants)*, 12
T2K *(unit_convert)*, 32
Tdew2q *(vapour_press)*, 33
Tdew2RH *(vapour_press)*, 33
Tdew2w *(vapour_press)*, 33
Tdew_from_q *(vapour_press)*, 33
Tdew_from_w *(vapour_press)*, 33
theta, 31
theta_wet *(theta)*, 31
theta_wet_bolton *(theta)*, 31

unit_convert, 32
usda_sf, 33

vapour_press, 33
VIC_soilParam, 35

W2_MJ *(unit_convert)*, 32
w2ea *(vapour_press)*, 33
W2mm *(unit_convert)*, 32
w2q *(vapour_press)*, 33
Wcr_FT *(VIC_soilParam)*, 35
wilting_point *(SPAW)*, 30
wilting_point_salinity *(SPAW)*, 30
Wpwp_FT *(VIC_soilParam)*, 35
ws2ssd *(cal_ssd)*, 10

XAJ_calib, 36
XAJ_calib(), *37*
XAJ_predict, 37