Package ‘tealeaves’

October 14, 2022

Version 1.0.6

Date 2022-07-19

Title Solve for Leaf Temperature Using Energy Balance

Depends R (>= 3.5.0), units (>= 0.6.0)

Imports checkmate (>= 2.0.0), crayon (>= 1.3.0), dplyr (>= 1.0.0), furrr (>= 0.1.0), future (>= 1.10.0), glue (>= 1.3.0), magrittr (>= 1.5.0), methods (>= 3.5.0), purrr (>= 0.3.0), rlang (>= 0.4.0), stringr (>= 1.4.0)

Suggests covr, ggplot2, knitr, rmarkdown, testthat, tidyr

Description

Implements models of leaf temperature using energy balance. It uses units to ensure that parameters are properly specified and transformed before calculations. It allows separate lower and upper surface conductances to heat and water vapour, so sensible and latent heat loss are calculated for each surface separately as in Foster and Smith (1986) <doi:10.1111/j.1365-3040.1986.tb02108.x>. It's straightforward to model leaf temperature over environmental gradients such as light, air temperature, humidity, and wind. It can also model leaf temperature over trait gradients such as leaf size or stomatal conductance. Other references are Monteith and Unsworth (2013, ISBN:9780123869104), Nobel (2009, ISBN:9780123741431), and Okajima et al. (2012) <doi:10.1007/s11284-011-0905-5>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

VignetteBuilder knitr

NeedsCompilation no

Author Chris Muir [aut, cre] (<https://orcid.org/0000-0003-2555-3878>)

Maintainer Chris Muir <cdmuir@hawaii.edu>

Repository CRAN

Date/Publication 2022-07-20 14:30:02 UTC

R topics documented:

`.get_dwv` *d_wv: water vapour gradient (mol / m ^ 3)*

Description

d_wv: water vapour gradient (mol / m ^ 3)

Usage

```
.get_dwv(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

Water vapour gradient: The water vapour pressure differential from inside to outside of the leaf is the saturation water vapor pressure inside the leaf (p_leaf) minus the water vapor pressure of the air (p_air):

$$d_{\mathrm{wv}} = p_{\mathrm{leaf}}/(RT_{\mathrm{leaf}}) - RHp_{\mathrm{air}}/(RT_{\mathrm{air}})$$

Note that water vapor pressure is converted from kPa to mol / m^3 using ideal gas law.

Symbol	*R*	*Description*	*Units*	*Default*
p_{air}	`p_air`	saturation water vapour pressure of air	kPa	calculated
p_{leaf}	`p_leaf`	saturation water vapour pressure inside the leaf	kPa	calculated
R	`R`	ideal gas constant	J / (mol K)	8.3144598
RH	`RH`	relative humidity	%	0.50
T_{air}	`T_air`	air temperature	K	298.15
T_{leaf}	`T_leaf`	leaf temperature	K	input

Value

Value in mol / m^3 of class `units`

Examples

```
# Water vapour gradient:

leaf_par <- make_leafpar()
enviro_par <- make_enviropar()
constants <- make_constants()
pars <- c(leaf_par, enviro_par, constants)
T_leaf <- set_units(300, K)
T_air <- set_units(298.15, K)
p_leaf <- set_units(35.31683, kPa)
p_air <- set_units(31.65367, kPa)

d_wv <- p_leaf / (pars$R * T_leaf) - pars$RH * p_air / (pars$R * T_air)
```

`.get_Dx` *D_x: Calculate diffusion coefficient for a given temperature and pressure*

Description

D_x: Calculate diffusion coefficient for a given temperature and pressure

Usage

```
.get_Dx(D_0, Temp, eT, P, unitless)
```

Arguments

`D_0`	Diffusion coefficient at 273.15 K (0 °C) and 101.3246 kPa
`Temp`	Temperature in Kelvin
`eT`	Exponent for temperature dependence of diffusion
`P`	Atmospheric pressure in kPa
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$D = D_0(T/273.15)^{eT}(101.3246/P)$$

According to Montieth & Unger (2013), eT is generally between 1.5 and 2. Their data in Appendix 3 indicate $eT = 1.75$ is reasonable for environmental physics.

Value

Value in m^2/s of class `units`

References

Monteith JL, Unsworth MH. 2013. Principles of Environmental Physics. 4th edition. Academic Press, London.

Examples

```
tealeaves:::.get_Dx(
  D_0 = set_units(2.12e-05, m^2/s),
  Temp = set_units(298.15, K),
  eT = set_units(1.75),
  P = set_units(101.3246, kPa),
```

```
  unitless = FALSE
)
```

.get_gbw *g_bw: Boundary layer conductance to water vapour (m / s)*

Description

g_bw: Boundary layer conductance to water vapour (m / s)

Usage

```
.get_gbw(T_leaf, surface, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`surface`	Leaf surface (lower or upper)
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$g_{\mathrm{bw}} = D_{\mathrm{w}} Sh/d$$

Symbol	*R*	*Description*	*Units*	*Default*
d	`leafsize`	Leaf characteristic dimension in meters	m	0.1
D_{w}	`D_w`	diffusion coefficient for water vapour	m^2 / s	calculated
Sh	`Sh`	Sherwood number	none	calculated

Value

Value in m / s of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()
```

```
T_leaf <- set_units(298.15, K)

tealeaves:::.get_gbw(T_leaf, "lower", c(cs, ep, lp), FALSE)
```

.get_gh *g_h: boundary layer conductance to heat (m / s)*

Description

g_h: boundary layer conductance to heat (m / s)

Usage

```
.get_gh(T_leaf, surface, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`surface`	Leaf surface (lower or upper)
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$g_\mathrm{h} = D_\mathrm{h} Nu / d$$

Symbol	*R*	*Description*	*Units*	*Default*
d	`leafsize`	Leaf characteristic dimension in meters	m	0.1
D_h	`D_h`	diffusion coefficient for heat in air	m^2 / s	calculated
Nu	`Nu`	Nusselt number	none	calculated

Value

Value in m/s of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()
```

```
T_leaf <- set_units(298.15, K)

tealeaves:::.get_gh(T_leaf, "lower", c(cs, ep, lp), FALSE)
```

`.get_gr` *Gr: Grashof number*

Description

Gr: Grashof number

Usage

```
.get_gr(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$Gr = t_{\mathrm{air}} G d^3 |T_{\mathrm{v,leaf}} - T_{\mathrm{v,air}}| / D_{\mathrm{m}}^2$$

Symbol	*R*	*Description*	*Units*	*Default*
d	`leafsize`	Leaf characteristic dimension in meters	m	0.1
D_{m}	`D_m`	diffusion coefficient of momentum in air	m^2 / s	calculated
G	`G`	gravitational acceleration	m / s^2	9.8
t_{air}	`t_air`	coefficient of thermal expansion of air	1 / K	1 / Temp
$T_{\mathrm{v,air}}$	`Tv_air`	virtual air temperature	K	calculated
$T_{\mathrm{v,leaf}}$	`Tv_leaf`	virtual leaf temperature	K	calculated

Value

A unitless number of class `units`

Examples

```
library(tealeaves)
```

```
cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_gr(T_leaf, c(cs, ep, lp), FALSE)
```

`.get_gtw` *g_tw: total conductance to water vapour (m/s)*

Description

g_tw: total conductance to water vapour (m/s)

Usage

```
.get_gtw(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

Total conductance to water vapor: The total conductance to water vapor (g_tw) is the sum of the parallel lower (abaxial) and upper (adaxial) conductances:

$$g_\mathrm{tw} = g_\mathrm{w,lower} + g_\mathrm{w,upper}$$

The conductance to water vapor on each surface is a function of parallel stomatal (g_sw) and cuticular (g_uw) conductances in series with the boundary layer conductance (g_bw). The stomatal, cuticular, and boundary layer conductance on the lower surface are:

$$g_\mathrm{sw,lower} = g_\mathrm{sw}(1 - sr)R(T_\mathrm{leaf} + T_\mathrm{air})/2$$

$$g_\mathrm{uw,lower} = g_\mathrm{uw}/2R(T_\mathrm{leaf} + T_\mathrm{air})/2$$

See `.get_gbw` for details on calculating boundary layer conductance. The equations for the upper surface are:

$$g_\mathrm{sw,upper} = g_\mathrm{sw} sr R(T_\mathrm{leaf} + T_\mathrm{air})/2$$

$$g_{\text{uw,upper}} = g_{\text{uw}}/2R(T_{\text{leaf}} + T_{\text{air}})/2$$

Note that the stomatal and cuticular conductances are given in units of (μmol H2O) / (m^2 s Pa) (see make_leafpar) and converted to m/s using the ideal gas law. The total leaf stomatal (g_{sw}) and cuticular (g_{uw}) conductances are partitioned across lower and upper surfaces. The stomatal conductance on each surface depends on stomatal ratio (sr); the cuticular conductance is assumed identical on both surfaces.

Symbol	*R*	*Description*	*Units*	*Default*
g_{sw}	`g_sw`	stomatal conductance to H2O	(μmol H2O) / (m^2 s Pa)	5
g_{uw}	`g_uw`	cuticular conductance to H2O	(μmol H2O) / (m^2 s Pa)	0.1
R	`R`	ideal gas constant	J / (mol K)	8.3144598
$\text{logit}(sr)$	`logit_sr`	stomatal ratio (logit transformed)	none	0 = logit(0.5)
T_{air}	`T_air`	air temperature	K	298.15
T_{leaf}	`T_leaf`	leaf temperature	K	input

Value

Value in m/s of class `units`

Examples

```
# Total conductance to water vapor

## Hypostomatous leaf; default parameters
leaf_par <- make_leafpar(replace = list(logit_sr = set_units(-Inf)))
enviro_par <- make_enviropar()
constants <- make_constants()
pars <- c(leaf_par, enviro_par, constants)
T_leaf <- set_units(300, K)

## Fixing boundary layer conductance rather than calculating
gbw_lower <- set_units(0.1, m / s)
gbw_upper <- set_units(0.1, m / s)

# Lower surface ----
## Note that pars$logit_sr is logit-transformed! Use stats::plogis() to convert to proportion.
gsw_lower <- set_units(pars$g_sw * (set_units(1) - stats::plogis(pars$logit_sr)) * pars$R *
                         ((T_leaf + pars$T_air) / 2), "m / s")
guw_lower <- set_units(pars$g_uw * 0.5 * pars$R * ((T_leaf + pars$T_air) / 2), m / s)
gtw_lower <- 1 / (1 / (gsw_lower + guw_lower) + 1 / gbw_lower)

# Upper surface ----
gsw_upper <- set_units(pars$g_sw * stats::plogis(pars$logit_sr) * pars$R *
                         ((T_leaf + pars$T_air) / 2), m / s)
guw_upper <- set_units(pars$g_uw * 0.5 * pars$R * ((T_leaf + pars$T_air) / 2), m / s)
gtw_upper <- 1 / (1 / (gsw_upper + guw_upper) + 1 / gbw_upper)

## Lower and upper surface are in parallel
g_tw <- gtw_lower + gtw_upper
```

.get_H — *H: sensible heat flux density (W / m^2)*

Description

H: sensible heat flux density (W / m^2)

Usage

```
.get_H(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$H = P_\mathrm{a} c_p g_\mathrm{h} (T_\mathrm{leaf} - T_\mathrm{air})$$

Symbol	*R*	*Description*	*Units*	*Default*
c_p	`c_p`	heat capacity of air	J / (g K)	1.01
g_h	`g_h`	boundary layer conductance to heat	m / s	calculated
P_a	`P_a`	density of dry air	g / m^3	calculated
T_air	`T_air`	air temperature	K	298.15
T_leaf	`T_leaf`	leaf temperature	K	input

Value

Value in W / m^2 of class `units`

See Also

`.get_gh`, `.get_Pa`

Examples

```
library(tealeaves)

cs <- make_constants()
```

```
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_H(T_leaf, c(cs, ep, lp), FALSE)
```

`.get_hvap` *h_vap: heat of vaporization (J / mol)*

Description

h_vap: heat of vaporization (J / mol)

Usage

```
.get_hvap(T_leaf, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

Heat of vaporization: The heat of vaporization (h_{vap}) is a function of temperature. I used data from on temperature and h_{vap} from Nobel (2009, Appendix 1) to estimate a linear regression. See Examples.

Value

Value in J/mol of class `units`

References

Nobel PS. 2009. Physicochemical and Environmental Plant Physiology. 4th Edition. Academic Press.

Examples

```
# Heat of vaporization and temperature
## data from Nobel (2009)

T_K <- 273.15 + c(0, 10, 20, 25, 30, 40, 50, 60)
h_vap <- 1e3 * c(45.06, 44.63, 44.21, 44.00,
```

```
                 43.78, 43.35, 42.91, 42.47) # (in J / mol)

fit <- lm(h_vap ~ T_K)

## coefficients are 56847.68250 J / mol and 43.12514 J / (mol K)

coef(fit)

T_leaf <- 298.15
h_vap <- set_units(56847.68250, J / mol) -
          set_units(43.12514, J / mol / K) * set_units(T_leaf, K)

## h_vap at 298.15 K is 43989.92 [J/mol]

set_units(h_vap, J / mol)

tealeaves:::.get_hvap(set_units(298.15, K), FALSE)
```

`.get_L` *L: Latent heat flux density (W / m^2)*

Description

L: Latent heat flux density (W / m^2)

Usage

```
.get_L(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$L = h_{\text{vap}} g_{\text{tw}} d_{\text{wv}}$$

Symbol	*R*	*Description*	*Units*	*Default*
d_{wv}	`d_wv`	water vapour gradient	mol / m ^ 3	calculated
h_{vap}	`h_vap`	latent heat of vaporization	J / mol	calculated
g_{tw}	`g_tw`	total conductance to H2O	(μmol H2O) / (m^2 s Pa)	calculated

Value

Value in W / m^2 of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_L(T_leaf, c(cs, ep, lp), FALSE)
```

`.get_nu` *Nu: Nusselt number*

Description

Nu: Nusselt number

Usage

```
.get_nu(T_leaf, surface, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`surface`	Leaf surface (lower or upper)
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

The Nusselt number depends on a combination how much free or forced convection predominates. For mixed convection:

$$Nu = (aRe^b)^{3.5} + (cGr^d)^{3.5})^{1/3.5}$$

Symbol	*R*	*Description*	*Units*	*Default*
a, b, c, d	`a, b, c, d`	empirical coefficients	none	calculated
Gr	`Gr`	Grashof number	none	calculated
Re	`Re`	Reynolds number	none	calculated

Value

A unitless number of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_nu(T_leaf, "lower", c(cs, ep, lp), FALSE)
```

`.get_Pa` *P_a: density of dry air (g / m^3)*

Description

P_a: density of dry air (g / m^3)

Usage

```
.get_Pa(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$P_{\mathrm{a}} = P/(R_{\mathrm{air}}(T_{\mathrm{leaf}} - T_{\mathrm{air}})/2)$$

Symbol	*R*	*Description*	*Units*	*Default*
P	`P`	atmospheric pressure	kPa	101.3246
R_{air}	`R_air`	specific gas constant for dry air	J / (kg K)	287.058
T_{air}	`T_air`	air temperature	K	298.15
T_{leaf}	`T_leaf`	leaf temperature	K	input

Value

Value in g / m^3 of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_Pa(T_leaf, c(cs, ep, lp), FALSE)
```

`.get_ps` *Saturation water vapour pressure (kPa)*

Description

Saturation water vapour pressure (kPa)

Usage

```
.get_ps(Temp, P, unitless)
```

Arguments

`Temp`	Temperature in Kelvin
`P`	Atmospheric pressure in kPa
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

Goff-Gratch equation (see http://cires1.colorado.edu/~voemel/vp.html)

This equation assumes P = 1 atm = 101.3246 kPa, otherwise boiling temperature needs to change

Value

Value in kPa of class `units`

References

http://cires1.colorado.edu/~voemel/vp.html

Examples

```
T_leaf <- set_units(298.15, K)
P <- set_units(101.3246, kPa)
tealeaves:::.get_ps(T_leaf, P, FALSE)
```

.get_Rabs *R_abs: total absorbed radiation (W / m^2)*

Description

R_abs: total absorbed radiation (W / m^2)

Usage

```
.get_Rabs(pars, unitless)
```

Arguments

pars	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
unitless	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

The following treatment follows Okajima et al. (2012):

$$R_{\text{abs}} = \alpha_{\text{s}}(1 + r)S_{\text{sw}} + \alpha_{\text{l}}\sigma(T_{\text{sky}}^4 + T_{\text{air}}^4)$$

The incident longwave (aka thermal infrared) radiation is modeled from sky and air temperature $\sigma(T_{\text{sky}}^4 + T_{\text{air}}^4)$ where T_{sky} is function of the air temperature and incoming solar shortwave radiation:

$$T_{\text{sky}} = T_{\text{air}} - 20S_{\text{sw}}/1000$$

Symbol	*R*	*Description*	*Units*	*Default*
α_{s}	`abs_s`	absorbtivity of shortwave radiation (0.3 - 4 μm)	none	0.80

α_{l}	`abs_l`	absorbtivity of longwave radiation (4 - 80 μm)	none	0.97
r	`r`	reflectance for shortwave irradiance (albedo)	none	0.2
σ	`s`	Stefan-Boltzmann constant	W / (m^2 K^4)	5.67e-08
S_{sw}	`S_sw`	incident short-wave (solar) radiation flux density	W / m^2	1000
S_{lw}	`S_lw`	incident long-wave radiation flux density	W / m^2	calculated
T_{air}	`T_air`	air temperature	K	298.15
T_{sky}	`T_sky`	sky temperature	K	calculated

Value

Value in W / m^2 of class `units`

References

Okajima Y, H Taneda, K Noguchi, I Terashima. 2012. Optimum leaf size predicted by a novel leaf energy balance model incorporating dependencies of photosynthesis on light and temperature. Ecological Research 27: 333-46.

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()
ep$T_sky <- ep$T_sky(ep)

tealeaves:::.get_Rabs(c(cs, ep, lp), FALSE)
```

`.get_re` *Re: Reynolds number*

Description

Re: Reynolds number

Usage

```
.get_re(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$Re = ud/D_{\mathrm{m}}$$

Symbol	*R*	*Description*	*Units*	*Default*
d	`leafsize`	Leaf characteristic dimension in meters	m	0.1
D_{m}	`D_m`	diffusion coefficient of momentum in air	m^2 / s	calculated
u	`wind`	windspeed	m / s	2

Value

A unitless number of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_re(T_leaf, c(cs, ep, lp), FALSE)
```

`.get_sh` *Sh: Sherwood number*

Description

Sh: Sherwood number

Usage

```
.get_sh(T_leaf, surface, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`surface`	Leaf surface (lower or upper)
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

The Sherwood number depends on a combination how much free or forced convection predominates. For mixed convection:

$$Sh = (aRe^b)^{3.5} + (cGr^d)^{3.5})^{1/3.5}$$

Symbol	*R*	*Description*	*Units*	*Default*
a, b, c, d	`a, b, c, d`	empirical coefficients	none	calculated
Gr	`Gr`	Grashof number	none	calculated
Re	`Re`	Reynolds number	none	calculated

Value

A unitless number of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_sh(T_leaf, "lower", c(cs, ep, lp), FALSE)
```

`.get_Sr` *S_r: longwave re-radiation (W / m^2)*

Description

S_r: longwave re-radiation (W / m^2)

Usage

```
.get_Sr(T_leaf, pars)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)

Details

$$S_{\mathrm{r}} = 2\sigma\alpha_{\mathrm{l}}T_{\mathrm{air}}^4$$

The factor of 2 accounts for re-radiation from both leaf surfaces (Foster and Smith 1986).

Symbol	*R*	*Description*	*Units*	*Default*
α_{l}	`abs_l`	absorbtivity of longwave radiation (4 - 80 μm)	none	0.97
T_{air}	`T_air`	air temperature	K	298.15
σ	`s`	Stefan-Boltzmann constant	W / (m^2 K^4)	5.67e-08

Note that leaf absorbtivity is the same value as leaf emissivity

Value

Value in W / m^2 of class `units`

References

Foster JR, Smith WK. 1986. Influence of stomatal distribution on transpiration in low-wind environments. Plant, Cell & Environment 9: 751-9.

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

tealeaves:::.get_Sr(T_leaf, c(cs, ep, lp))
```

`.get_Tv` *Calculate virtual temperature*

Description

Calculate virtual temperature

Usage

```
.get_Tv(Temp, p, P, epsilon, unitless)
```

Arguments

Temp	Temperature in Kelvin
p	water vapour pressure in kPa
P	Atmospheric pressure in kPa
epsilon	ratio of water to air molar masses (unitless)
unitless	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

$$T_\mathrm{v} = T/[1 - (1 - \epsilon)(p/P)]$$

Eq. 2.35 in Monteith & Unsworth (2013)

Symbol	*R*	*Description*	*Units*	*Default*
ϵ	`epsilon`	ratio of water to air molar masses	unitless	0.622
p	`p`	water vapour pressure	kPa	calculated
P	`P`	atmospheric pressure	kPa	101.3246

Value

Value in K of class `units`

References

Monteith JL, Unsworth MH. 2013. Principles of Environmental Physics. 4th edition. Academic Press, London.

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)
p <- ep$RH * tealeaves:::.get_ps(T_leaf, ep$P, FALSE)
tealeaves:::.get_Tv(T_leaf, p, ep$P, cs$epsilon, FALSE)
```

Ar *Ar: Archimedes number*

Description

Ar: Archimedes number

Usage

```
Ar(T_leaf, pars, unitless = FALSE)
```

Arguments

T_leaf	Leaf temperature in Kelvin
pars	Concatenated parameters (leaf_par, enviro_par, and constants)
unitless	Logical. Should function use parameters with units? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

The Archimedes number is a dimensionless number that describes when free or forced convection dominates.

$$Ar = Gr/Re^2$$

Symbol	*R*	*Description*	*Units*	*Default*
Gr	Gr	Grashof number	none	calculated
Re	Re	Reynolds number	none	calculated

Value

unitless = TRUE: A unitless number of class numeric unitless = FALSE: A unitless number of class units Also returns Reynolds and Grashof numbers

Examples

```
cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()
pars <- c(cs, lp, ep)
T_leaf <- set_units(298.15, "K")

Ar(T_leaf, pars)
```

constants — *S3 class constants*

Description

Constructor function for constants class. This function ensures that physical constant inputs are properly formatted.

Usage

```
constants(.x)
```

Arguments

`.x`	A list to be constructed into **constants**. If units are not provided, they will be set without conversion. If units are provided, they will be checked and converted to units that tealeaves uses.

convert_conductance — *Convert conductance units*

Description

Convert conductance units

Usage

```
convert_conductance(.g, Temp = NULL, P = NULL)
```

Arguments

`.g`	Conductance in class units. Units must convertible to one of "m/s", "umol/m^2/s/Pa", or "mol/m^2/s"
`Temp`	A temperature value of class `units`
`P`	A pressure value of class `units` that is convertible to kPa

Value

A list of three values of clas `units` with units "m/s", "umol/m^2/s/Pa", and "mol/m^2/s".

Examples

```
g_sw <- set_units(10, "m/s")
convert_conductance(g_sw,
                    Temp = set_units(298.15, "K"),
                    P = set_units(101.3246, "kPa"))

g_sw <- set_units(4, "umol/m^2/s/Pa")
convert_conductance(g_sw,
                    Temp = set_units(298.15, "K"),
                    P = set_units(101.3246, "kPa"))

g_sw <- set_units(0.4, "mol/m^2/s")
convert_conductance(g_sw,
                    Temp = set_units(298.15, "K"),
                    P = set_units(101.3246, "kPa"))
```

E *Evaporation (mol / (m^2 s))*

Description

Evaporation (mol / (m^2 s))

Usage

```
E(T_leaf, pars, unitless)
```

Arguments

`T_leaf`	Leaf temperature in Kelvin
`pars`	Concatenated parameters (`leaf_par`, `enviro_par`, and `constants`)
`unitless`	Logical. Should function use parameters with `units`? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Details

The leaf evaporation rate is the product of the total conductance to water vapour (m / s) and the water vapour gradient (mol / m^3):

$$E = g_{\mathrm{tw}} D_{\mathrm{wv}}$$

If `unitless = TRUE`, `T_leaf` is assumed in degrees K without checking.

Value

`unitless` = `TRUE`: A value in units of mol / (m ^ 2 / s) number of class `numeric` `unitless` = `FALSE`: A value in units of mol / (m ^ 2 / s) of class `units`

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()

T_leaf <- set_units(298.15, K)

E(T_leaf, c(cs, ep, lp), FALSE)
```

`energy_balance` *Calculate leaf energy balance*

Description

Calculate leaf energy balance

Usage

```
energy_balance(
  tleaf,
  leaf_par,
  enviro_par,
  constants,
  quiet = FALSE,
  components = FALSE,
  set_units = FALSE
)
```

Arguments

`tleaf`	Leaf temperature in Kelvin. If input is numeric, it will be automatically converted to `units`.
`leaf_par`	A list of leaf parameters. This can be generated using the `make_leafpar` function.
`enviro_par`	A list of environmental parameters. This can be generated using the `make_enviropar` function.
`constants`	A list of physical constants. This can be generated using the `make_constants` function.

quiet	Logical. Should a message appear about conversion from `numeric` to `units`? Useful for finding leaf temperature that balances heat transfer using `uniroot`.
components	Logical. Should leaf energy components be returned? Transpiration (in mol / (m^2 s)) also returned.
set_units	Logical. Should `units` be set? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.

Value

A numeric value in W / m^2. Optionally, a named list of energy balance components in W / m^2 and transpiration in mol / (m^2 s).

Examples

```
library(tealeaves)

cs <- make_constants()
ep <- make_enviropar()
lp <- make_leafpar()
ep$T_sky <- ep$T_sky(ep)

T_leaf <- set_units(298.15, K)

energy_balance(T_leaf, lp, ep, cs, FALSE, TRUE, TRUE)
```

`enviro_par` *S3 class enviro_par*

Description

Constructor function for enviro_par class. This function ensures that environmental parameter inputs are properly formatted.

Usage

```
enviro_par(.x)
```

Arguments

.x	A list to be constructed into **enviro_par**. If units are not provided, they will be set without conversion. If units are provided, they will be checked and converted to units that tealeaves uses.

`leaf_par` *S3 class leaf_par*

Description

Constructor function for leaf_par class. This function ensures that leaf parameter inputs are properly formatted.

Usage

```
leaf_par(.x)
```

Arguments

`.x`	A list to be constructed into **leaf_par**. If units are not provided, they will be set without conversion. If units are provided, they will be checked and converted to units that tealeaves uses.

`make_parameters` *Make lists of parameters of leaf, environmental, or constant parameters*

Description

Make lists of parameters of leaf, environmental, or constant parameters

make_leafpar

make_enviropar

make_constants

Usage

```
make_leafpar(replace = NULL)

make_enviropar(replace = NULL)

make_constants(replace = NULL)
```

Arguments

`replace`	A named list of parameters to replace defaults. If `NULL`, defaults will be used.

Details

Leaf parameters:

Symbol	*R*	*Description*	*Units*	*Default*
d	`leafsize`	Leaf characteristic dimension	m	0.1
α_l	`abs_l`	absorbtivity of longwave radiation (4 - 80 μm)	none	0.97
α_s	`abs_s`	absorbtivity of shortwave radiation (0.3 - 4 μm)	none	0.50
g_sw	`g_sw`	stomatal conductance to H2O	(μmol H2O) / (m^2 s Pa)	5
g_uw	`g_uw`	cuticular conductance to H2O	(μmol H2O) / (m^2 s Pa)	0.1
$\mathrm{logit}(sr)$	`logit_sr`	stomatal ratio (logit transformed)	none	0 = logit(0.5)

Environment parameters:

Symbol	*R*	*Description*	*Units*	*Default*
P	`P`	atmospheric pressure	kPa	101.3246
r	`r`	reflectance for shortwave irradiance (albedo)	none	0.2
RH	`RH`	relative humidity	none	0.50
S_sw	`S_sw`	incident short-wave (solar) radiation flux density	W / m^2	1000
S_lw	`S_lw`	incident long-wave radiation flux density	W / m^2	calculated
T_air	`T_air`	air temperature	K	298.15
u	`wind`	windspeed	m / s	2

Constants:

Symbol	*R*	*Description*	*Units*	*Default*
c_p	`c_p`	heat capacity of air	J / (g K)	1.01
$D_{h,0}$	`D_h0`	diffusion coefficient for heat in air at 0 °C	m^2 / s	19.0e-06
$D_{m,0}$	`D_m0`	diffusion coefficient for momentum in air at 0 °C	m^2 / s	13.3e-06
$D_{w,0}$	`D_w0`	diffusion coefficient for water vapour in air at 0 C	m^2 / s	21.2e-06
ϵ	`epsilon`	ratio of water to air molar masses	none	0.622
eT	`eT`	exponent for temperature dependence of diffusion	none	1.75
G	`G`	gravitational acceleration	m / s^2	9.8
Nu	`Nu`	Nusselt number	none	calculated
R	`R`	ideal gas constant	J / (mol K)	8.3144598
R_air	`R_air`	specific gas constant for dry air	J / (kg K)	287.058
σ	`s`	Stefan-Boltzmann constant	W / (m^2 K^4)	5.67e-08
Sh	`Sh`	Sherwood number	none	calculated

Value

`make_leafpar`: An object inheriting from class `leaf_par`
`make_enviropar`: An object inheriting from class `enviro_par`
`make_constants`: An object inheriting from class `constants`

Examples

```
library(tealeaves)

# Use defaults
cs <- make_constants()
ep <- make_enviropar()
```

```
lp <- make_leafpar()

# Replace defaults

ep <- make_enviropar(
  replace = list(
    T_air = set_units(300, K)
  )
)

lp <- make_leafpar(
  replace = list(
    leafsize = set_units(c(0.1, 0.2), m)
  )
)
```

`parameter_names` *Get vector of parameter names*

Description

Get vector of parameter names

Usage

```
parameter_names(which)
```

Arguments

`which`	A character string indicating which parameter names to retrieve, "constants", "enviro", or "leaf". Partial matching allowed.

Examples

```
parameter_names("leaf")
```

`tealeaves` `tealeaves` *package*

Description

Solve for Leaf Temperature Using Energy Balance

Details

See the README on GitHub

`tleaves`	`tleaves`: *find leaf temperatures for multiple parameter sets*

Description

`tleaves`: find leaf temperatures for multiple parameter sets

`tleaf`: find leaf temperatures for a single parameter set

Usage

```
tleaves(
  leaf_par,
  enviro_par,
  constants,
  progress = TRUE,
  quiet = FALSE,
  set_units = TRUE,
  parallel = FALSE
)

tleaf(leaf_par, enviro_par, constants, quiet = FALSE, set_units = TRUE)
```

Arguments

`leaf_par`	A list of leaf parameters. This can be generated using the `make_leafpar` function.
`enviro_par`	A list of environmental parameters. This can be generated using the `make_enviropar` function.
`constants`	A list of physical constants. This can be generated using the `make_constants` function.
`progress`	Logical. Should a progress bar be displayed?
`quiet`	Logical. Should messages be displayed?
`set_units`	Logical. Should `units` be set? The function is faster when FALSE, but input must be in correct units or else results will be incorrect without any warning.
`parallel`	Logical. Should parallel processing be used via `future_map`?

Value

`tleaves`:

A tibble with the following `units` columns

Input:

`abs_l`	Absorbtivity of longwave radiation (unitless)

abs_s	Absorbtivity of shortwave radiation (unitless)
g_sw	Stomatal conductance to H2O (μmol H2O / (m^2 s Pa))
g_uw	Cuticular conductance to H2O (μmol H2O / (m^2 s Pa))
leafsize Leaf characteristic dimension	(m)
logit_sr	Stomatal ratio (logit transformed; unitless)
P	Atmospheric pressure (kPa)
RH	Relative humidity (unitless)
S_lw	incident long-wave radiation flux density (W / m^2)
S_sw	incident short-wave (solar) radiation flux density (W / m^2)
T_air	Air temperature (K)
wind	Wind speed (m / s)
Output:	
T_leaf	Equilibrium leaf tempearture (K)
value	Leaf energy balance (W / m^2) at tleaf
convergence	Convergence code (0 = converged)
R_abs	Total absorbed radiation (W / m^2; see .get_Rabs)
S_r	Thermal infrared radiation loss (W / m^2; see .get_Sr)
H	Sensible heat flux density (W / m^2; see .get_H)
L	Latent heat flux density (W / m^2; see .get_L)
E	Evapotranspiration (mol H2O/ (m^2 s))

tleaf:

A data.frame with the following numeric columns:

T_leaf	Equilibrium leaf temperature (K)
value	Leaf energy balance (W / m^2) at tleaf
convergence	Convergence code (0 = converged)
R_abs	Total absorbed radiation (W / m^2; see .get_Rabs)
S_r	Longwave re-radiation (W / m^2; see .get_Sr)
H	Sensible heat flux density (W / m^2; see .get_H)
L	Latent heat flux density (W / m^2; see .get_L)
E	Evapotranspiration (mol H2O/ (m^2 s))

Examples

```
# tleaf for single parameter set:

leaf_par <- make_leafpar()
enviro_par <- make_enviropar()
constants <- make_constants()
tleaf(leaf_par, enviro_par, constants)

# tleaves for multiple parameter set:

enviro_par <- make_enviropar(
  replace = list(
    T_air = set_units(c(293.15, 298.15), K)
```

```
  )
)
tleaves(leaf_par, enviro_par, constants)
```

`tl_example1` *tealeaves example output 1*

Description

An example output from the `tealeaves` function.

Usage

```
tl_example1
```

Format

A data frame with 150 rows and 20 variables:

Index