

Package ‘tealeaves’

February 6, 2026

Version 1.0.7

Date 2026-02-05

Title Solve for Leaf Temperature Using Energy Balance

Depends R ($\geq 4.1.0$), units ($\geq 0.6.0$)

Imports checkmate ($\geq 2.0.0$), crayon ($\geq 1.3.0$), dplyr ($\geq 1.0.0$),
furr ($\geq 0.1.0$), glue ($\geq 1.3.0$), magrittr ($\geq 1.5.0$), methods
($\geq 3.5.0$), progressr ($\geq 0.18.0$), purrr ($\geq 0.3.0$), rlang ($\geq$
0.4.0), stringr ($\geq 1.4.0$)

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](https://doi.org/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](https://doi.org/10.1007/s11284-011-0905-5)>.

License MIT + file LICENSE

URL <https://github.com/cdmuir/tealeaves>,
<https://cdmuir.github.io/tealeaves/>

BugReports <https://github.com/cdmuir/tealeaves/issues>

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

VignetteBuilder knitr

NeedsCompilation no

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

Maintainer Chris Muir <cdmuir@wisc.edu>

Repository CRAN
Date/Publication 2026-02-06 11:10:02 UTC

Contents

.get_dwv	2
.get_Dx	4
.get_gbw	5
.get_gh	6
.get_gr	7
.get_gtw	8
.get_H	9
.get_hvap	11
.get_L	12
.get_nu	13
.get_Pa	14
.get_ps	15
.get_Rabs	16
.get_re	17
.get_sh	18
.get_Sr	19
.get_Tv	20
Ar	22
constants	23
convert_conductance	23
E	24
energy_balance	25
enviro_par	26
leaf_par	27
make_parameters	27
parameter_names	29
tleaves	30
tl_example1	32
Index	33

.get_dwv	<i>d_wv: water vapour gradient (mol / m ^ 3)</i>
----------	--

Description

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

Usage

.get_dwv(T_leaf, pars, unitless)

Arguments

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>pars</code>	Concatenated parameters (<code>leaf_par</code> , <code>enviro_par</code> , and constants)
<code>unitless</code>	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`):

$$d_{wv} = p_{\text{leaf}} / (RT_{\text{leaf}}) - RH p_{\text{air}} / (RT_{\text{air}})$$

Note that water vapor pressure is converted from kPa to mol / m³ using ideal gas law.

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

Value

Value in mol / m³ 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	<i>D_x: Calculate diffusion coefficient for a given temperature and pressure</i>
---------	--

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 Monteith & 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 m²/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

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>surface</code>	Leaf surface (lower or upper)
<code>pars</code>	Concatenated parameters (leaf_par, enviro_par, and constants)
<code>unitless</code>	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

$$g_{bw} = D_w Sh / d$$

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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	<i>g_h: boundary layer conductance to heat (m / s)</i>
---------	--

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

$$g_h = D_h Nu / d$$

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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)
```

<code>.get_gr</code>	<i>Gr: Grashof number</i>
----------------------	---------------------------

Description

Gr: Grashof number

Usage`.get_gr(T_leaf, pars, unitless)`**Arguments**

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>pars</code>	Concatenated parameters (leaf_par, enviro_par, and constants)
<code>unitless</code>	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

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

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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
$T_{\text{v,air}}$	Tv_air	virtual air temperature	K	calculated
$T_{\text{v,leaf}}$	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	<i>g_tw: total conductance to water vapour (m/s)</i>
----------	--

Description

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

Usage

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

Arguments

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>pars</code>	Concatenated parameters (<code>leaf_par</code> , <code>enviro_par</code> , and constants)
<code>unitless</code>	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:

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

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:

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

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

See [.get_gbw](#) for details on calculating boundary layer conductance. The equations for the upper surface are:

$$g_{sw,upper} = g_{sw}srR(T_{leaf} + T_{air})/2$$

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

Note that the stomatal and cuticular conductances are given in units of ($\mu\text{mol H}_2\text{O}$) / ($\text{m}^2 \text{ s Pa}$) (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.

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
g_{sw}	g_sw	stomatal conductance to H2O	$(\mu\text{mol H}_2\text{O}) / (\text{m}^2 \text{ s Pa})$	5
g_{uw}	g_uw	cuticular conductance to H2O	$(\mu\text{mol H}_2\text{O}) / (\text{m}^2 \text{ s Pa})$	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

$$H = P_a c_p g_h (T_{\text{leaf}} - T_{\text{air}})$$

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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 / m² 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)
```

<code>.get_hvap</code>	<i>h_{vap}: heat of vaporization (J / mol)</i>
------------------------	--

Description

h_{vap} : heat of vaporization (J / mol)

Usage

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

Arguments

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>unitless</code>	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	<i>L: Latent heat flux density (W / m^2)</i>
--------	--

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

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

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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	<i>Nu: Nusselt number</i>
---------	---------------------------

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:

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

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
<i>a, b, c, d</i>	a, b, c, d	empirical coefficients	none	calculated
<i>Gr</i>	Gr	Grashof number	none	calculated
<i>Re</i>	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	<i>P_a: density of dry air (g / m³)</i>
---------	--

Description

P_a: density of dry air (g / m³)

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

$$P_a = P / (R_{\text{air}}(T_{\text{leaf}} - T_{\text{air}}) / 2)$$

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
<i>P</i>	P	atmospheric pressure	kPa	101.3246
<i>R_{air}</i>	R _{air}	specific gas constant for dry air	J / (kg K)	287.058
<i>T_{air}</i>	T _{air}	air temperature	K	298.15
<i>T_{leaf}</i>	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 <https://cires1.colorado.edu/~voemel/vp.html>)

This equation assumes $P = 1 \text{ atm} = 101.3246 \text{ kPa}$, otherwise boiling temperature needs to change

Value

Value in kPa of class units

References

<https://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	<i>R_abs: total absorbed radiation (W / m^2)</i>
-----------	--

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):

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

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:

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

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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}	<code>T_air</code>	air temperature	K	298.15
T_{sky}	<code>T_sky</code>	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)
```

<code>.get_re</code>	<i>Re: Reynolds number</i>
----------------------	----------------------------

Description

Re: Reynolds number

Usage

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

Arguments

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>pars</code>	Concatenated parameters (leaf_par, enviro_par, and constants)
<code>unitless</code>	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

$$Re = ud/D_m$$

Symbol	R	Description	Units	Default
<i>d</i>	leafsize	Leaf characteristic dimension in meters	m	0.1
<i>D_m</i>	D_m	diffusion coefficient of momentum in air	m ² / s	calculated
<i>u</i>	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	<i>Sh: Sherwood number</i>
---------	----------------------------

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:

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

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
<i>a, b, c, d</i>	a, b, c, d	empirical coefficients	none	calculated
<i>Gr</i>	Gr	Grashof number	none	calculated
<i>Re</i>	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

<code>T_leaf</code>	Leaf temperature in Kelvin
<code>pars</code>	Concatenated parameters (leaf_par, enviro_par, and constants)

Details

$$S_r = 2\sigma\alpha_lT_{air}^4$$

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

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
α_l	abs_l	absorbtivity of longwave radiation (4 - 80 μ m)	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 / m² 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	<i>Calculate virtual temperature</i>
---------	--------------------------------------

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

$$T_v = T/[1 - (1 - \epsilon)(p/P)]$$

Eq. 2.35 in Monteith & Unsworth (2013)

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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	<i>Constants</i>
-----------	------------------

Description

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.

Details

Constructor for the S3 class "constants".

convert_conductance	<i>Convert conductance units</i>
---------------------	----------------------------------

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 class 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	<i>Evaporation (mol / (m^2 s))</i>
---	------------------------------------

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):

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

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	<i>Calculate leaf energy balance</i>
----------------	--------------------------------------

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 $\text{mol} / (\text{m}^2 \text{ s})$.

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

Environmental parameters

Description

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.
----	---

Details

Constructor for the S3 class "enviro_par".

leaf_par	<i>Leaf parameters</i>
----------	------------------------

Description

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.
----	---

Details

Constructor for the S3 class "leaf_par".

make_parameters	<i>Make lists of parameters of leaf, environmental, or constant parameters</i>
-----------------	--

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:

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
d	leafsize	Leaf characteristic dimension	m	0.1
α_l	abs_l	absorbitivity of longwave radiation (4 - 80 μm)	none	0.97
α_s	abs_s	absorbitivity of shortwave radiation (0.3 - 4 μm)	none	0.50
g_{sw}	g_sw	stomatal conductance to H2O	($\mu\text{mol H}_2\text{O}$) / ($\text{m}^2 \text{ s Pa}$)	5
g_{uw}	g_uw	cuticular conductance to H2O	($\mu\text{mol H}_2\text{O}$) / ($\text{m}^2 \text{ s Pa}$)	0.1
$\text{logit}(sr)$	logit_sr	stomatal ratio (logit transformed)	none	0 = logit(0.5)

Environment parameters:

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
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:

<i>Symbol</i>	<i>R</i>	<i>Description</i>	<i>Units</i>	<i>Default</i>
c_p	c_p	heat capacity of air	$\text{J} / (\text{g K})$	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	$\text{J} / (\text{mol K})$	8.3144598
R_{air}	R_air	specific gas constant for dry air	$\text{J} / (\text{kg K})$	287.058
σ	s	Stefan-Boltzmann constant	$\text{W} / (\text{m}^2 \text{ K}^4)$	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	<i>Get vector of parameter names</i>
-----------------	--------------------------------------

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")
```

tleaves	tleaves: <i>find leaf temperatures for multiple parameter sets</i>
---------	--

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 <code>make_leafpar</code> function.
enviro_par	A list of environmental parameters. This can be generated using the <code>make_enviropar</code> function.
constants	A list of physical constants. This can be generated using the <code>make_constants</code> 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 ($\mu\text{mol H}_2\text{O} / (\text{m}^2 \text{ s Pa})$)
g_uw	Cuticular conductance to H2O ($\mu\text{mol H}_2\text{O} / (\text{m}^2 \text{ s Pa})$)
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 ($\text{mol H}_2\text{O} / (\text{m}^2 \text{ s})$)

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 ($\text{mol H}_2\text{O} / (\text{m}^2 \text{ s})$)

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	<i>tealeaves example output 1</i>
-------------	-----------------------------------

Description

An example output from the [tealeaves](#) function.

Usage

```
tl_example1
```

Format

A data frame with 150 rows and 20 variables:

Index

- * **datasets**
 - tl_example1, 32
 - .get_Dx, 4
 - .get_H, 9, 31
 - .get_L, 12, 31
 - .get_Pa, 10, 14
 - .get_Rabs, 16, 31
 - .get_Sr, 19, 31
 - .get_Tv, 20
 - .get_dvw, 2
 - .get_gbw, 5, 8
 - .get_gh, 6, 10
 - .get_gr, 7
 - .get_gtw, 8
 - .get_hvap, 11
 - .get_nu, 13
 - .get_ps, 15
 - .get_re, 17
 - .get_sh, 18
- Ar, 22
- calculated, 3, 5–7, 10, 12, 13, 18, 19, 21, 22, 28
- constants, 23, 28
- convert_conductance, 23
- E, 24
- energy_balance, 25
- enviro_par, 26, 28
- future_map, 30
- leaf_par, 27, 28
- make_constants (make_parameters), 27
- make_enviropar (make_parameters), 27
- make_leafpar, 8
- make_leafpar (make_parameters), 27
- make_parameters, 27
- parameter_names, 29
- tealeaves, 32
- tl_example1, 32
- tleaf (tleaves), 30
- tleaves, 30
- uniroot, 25