Package ‘kgen’

December 19, 2023

Type Package

Title A Tool for Calculating Stoichiometric Equilibrium Constants (Ks) for Seawater

Version 0.3.1

License MIT + file LICENSE

Description A unified software package simultaneously implemented in 'Python', 'R', and 'Matlab' providing a uniform and internally-consistent way of calculating stoichiometric equilibrium constants in modern and palaeo seawater as a function of temperature, salinity, pressure and the concentration of magnesium, calcium, sulphate, and fluorine.

Encoding UTF-8

Depends R (>= 4.0)

Imports rjson (>= 0.2.21), reticulate (>= 1.26), rappdirs (>= 0.3.3), checkmate (>= 2.1.0), pbapply (>= 1.7.0), data.table (>= 1.14.6)

Suggests testthat (>= 3.0.0)

RoxygenNote 7.2.3

Config/testthat/edition 3

NeedsCompilation no

Author Dennis Mayk [aut, cre] (<https://orcid.org/0000-0002-5017-1495>)

Maintainer Dennis Mayk <dm807@cantab.ac.uk>

Repository CRAN

Date/Publication 2023-12-19 14:10:02 UTC

R topics documented:

`calc_fluorine` *Calculate fluorine*

Description

Calculate fluorine

Usage

```
calc_fluorine(sal)
```

Arguments

`sal`	Salinity

Value

fluorine

References

From Dickson et al., 2007, Table 2, Note: Sal / 1.80655 = Chlorinity

calc_ionic_strength *Ionic strength after Dickson (1990a); see Dickson et al. (2007)*

Description

Ionic strength after Dickson (1990a); see Dickson et al. (2007)

Usage

```
calc_ionic_strength(sal)
```

Arguments

sal	Salinity

Value

Ionic strength

calc_K *Calculate a single equilibrium constant*

Description

Calculate **a single** specified stoichiometric equilibrium constant at given temperature, salinity, pressure and the concentration of magnesium, calcium, sulphate, and fluorine.

Usage

```
calc_K(
  k,
  temp_c = 25,
  sal = 35,
  p_bar = NULL,
  magnesium = 0.0528171,
  calcium = 0.0102821,
  sulphate = NULL,
  fluorine = NULL,
  method = "r_polynomial"
)

calc_Ks(
  ks = NULL,
  temp_c = 25,
  sal = 35,
  p_bar = NULL,
  magnesium = 0.0528171,
  calcium = 0.0102821,
  sulphate = NULL,
  fluorine = NULL,
  method = "r_polynomial"
)

calc_all_Ks(
  temp_c = 25,
  sal = 35,
  p_bar = NULL,
  magnesium = 0.0528171,
  calcium = 0.0102821,
  sulphate = NULL,
  fluorine = NULL,
  method = "r_polynomial"
)
```

Arguments

`k` K to be calculated

`temp_c` Temperature (Celcius)

`sal` Salinity

`p_bar` Pressure (Bar) (optional)

`magnesium` magnesium concentration in mol/kgsw. If None, modern is assumed (0.0528171). Should be the average magnesium concentration in seawater - a salinity correction is then applied to calculate the magnesium concentration in the sample.

`calcium` calcium concentration in mol/kgsw. If None, modern is assumed (0.0102821). Should be the average calcium concentration in seawater - a salinity correction is then applied to calculate the magnesium concentration in the sample.

`sulphate` Sulphate concentration in mol/kgsw. Calculated from salinity if not given.

`fluorine` Fluorine concentration in mol/kgsw. Calculated from salinity if not given.

`method` string describing method which should be either 'myami', 'myami_polynomial', or 'r_polynomial' (Default: 'r_polynomial').

`ks` character vectors of Ks to be calculated e.g., c("K0", "K1") (Default: NULL, calculate all Ks)

Value

A single K at given conditions

Data.table of **multiple** Ks at given conditions

Data.table of **all** Ks at given conditions

Functions

- `calc_Ks()`: Wrapper to calculate **multiple** stoichiometric equilibrium constants at given temperature, salinity, pressure and the concentration of magnesium, calcium, sulphate, and fluorine.
- `calc_all_Ks()`: Wrapper to calculate **all** stoichiometric equilibrium constants at given temperature, salinity, pressure and the concentration of magnesium, calcium, sulphate, and fluorine.

Author(s)

Dennis Mayk

`calc_K0` *Calculate K0*

Description

Calculate K0

Usage

```
calc_K0(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

K0

`calc_K1` *Calculate K1*

Description

Calculate K1

Usage

```
calc_K1(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

K1

`calc_K2` *Calculate K2*

Description

Calculate K2

Usage

```
calc_K2(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

K2

`calc_KB` *Calculate KB*

Description

Calculate KB

Usage

```
calc_KB(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KB

`calc_KF` *Calculate KF*

Description

Calculate KF

Usage

```
calc_KF(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KF

`calc_KP1` *Calculate KP1*

Description

Calculate KP1

Usage

```
calc_KP1(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KP1

calc_KP2 *Calculate KP2*

Description

Calculate KP2

Usage

```
calc_KP2(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KP2

calc_KP3 *Calculate KP3*

Description

Calculate KP3

Usage

```
calc_KP3(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KP3

`calc_KS` *Calculate KS*

Description

Calculate KS

Usage

```
calc_KS(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KS

`calc_KSi` *Calculate KSi*

Description

Calculate KSi

Usage

```
calc_KSi(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KSi

`calc_Ksp` *Calculate Ksp*

Description

Calculate Ksp

Usage

```
calc_Ksp(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

Ksp

`calc_KW` *Calculate KW*

Description

Calculate KW

Usage

```
calc_KW(coefficients, temp_c, sal)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`sal`	Salinity

Value

KW

`calc_pc` *Calculate pressure correction factor for Ks*

Description

Calculate pressure correction factor for Ks

Usage

```
calc_pc(coefficients, temp_c, p_bar)
```

Arguments

`coefficients`	Coefficients for K calculation
`temp_c`	Temperature (Celcius)
`p_bar`	Pressure (Bar)

Value

Pressure correction factor

References

From Millero et al. (2007, doi:10.1021/cr0503557), Eqns 38-40

`calc_pressure_correction` *Calculate pressure correction factor*

Description

Calculate pressure correction factor for a specified equilibrium constant.

Usage

```
calc_pressure_correction(k, temp_c, p_bar)
```

Arguments

`k`	K to be calculated
`temp_c`	Temperature (Celcius)
`p_bar`	Pressure (Bar)

Value

pressure correction factor

Author(s)

Dennis Mayk

`calc_seawater_correction` *Kgen seawater composition correction function*

Description

Kgen seawater composition correction function

Usage

```
calc_seawater_correction(
  k,
  sal,
  temp_c,
  magnesium = 0.0528171,
  calcium = 0.0102821,
  method = "r_polynomial"
)
```

Arguments

`k`	K to be calculated
`sal`	Salinity
`temp_c`	Temperature (Celcius)
`magnesium`	magnesium concentration in mol/kgsw. If None, modern is assumed (0.0528171). Should be the average magnesium concentration in seawater - a salinity correction is then applied to calculate the magnesium concentration in the sample.
`calcium`	calcium concentration in mol/kgsw. If None, modern is assumed (0.0102821). Should be the average calcium concentration in seawater - a salinity correction is then applied to calculate the magnesium concentration in the sample.
`method`	string describing method which should be either 'myami', 'myami_polynomial', or 'r_polynomial' (Default: 'r_polynomial').

Value

list of seawater correction factors

Author(s)

Dennis Mayk

`calc_sulphate` *Calculate sulphate*

Description

Calculate sulphate

Usage

```
calc_sulphate(sal)
```

Arguments

`sal`	Salinity

Value

sulphate

References

From Dickson et al., 2007, Table 2, Note: Sal / 1.80655 = Chlorinity

`install_pymyami` *Install MyAMI from pypi*

Description

Function to install pymyami in a default location.

Usage

```
install_pymyami()
```

`is_linux` *Check if OS is Linux*

Description

Check if OS is Linux

Usage

```
is_linux()
```

is_osx *Check if OS is OSX*

Description

Check if OS is OSX

Usage

```
is_osx()
```

is_windows *Check if OS is Windows*

Description

Check if OS is Windows

Usage

```
is_windows()
```

kgen_poly *Kgen R polynomial function*

Description

Kgen R polynomial function

Usage

```
kgen_poly(sal, temp_c, magnesium = 0.0528171, calcium = 0.0102821)
```

Arguments

`sal`	Salinity
`temp_c`	Temperature (Celcius)
`magnesium`	magnesium concentration in mol/kgsw. If None, modern is assumed (0.0528171). Should be the average magnesium concentration in seawater - a salinity correction is then applied to calculate the magnesium concentration in the sample.
`calcium`	calcium concentration in mol/kgsw. If None, modern is assumed (0.0102821). Should be the average calcium concentration in seawater - a salinity correction is then applied to calculate the magnesium concentration in the sample.

Author(s)

Dennis Mayk

`K_fns` *List of all functions*

Description

List of all functions

Usage

```
K_fns
```

Format

An object of class `list` of length 13.

`mc_exists` *Check if miniconda exists*

Description

Check if miniconda exists

Usage

```
mc_exists(path = miniconda_path())
```

Arguments

`path`	Path to miniconda

`miniconda_conda` *Check if miniconda is installed*

Description

Check if miniconda is installed

Usage

```
miniconda_conda(path = miniconda_path())
```

Arguments

`path`	Path to miniconda

`miniconda_path` *Get miniconda path*

Description

Get miniconda path

Usage

```
miniconda_path()
```

`miniconda_path_default` *Get miniconda default path*

Description

Get miniconda default path

Usage

```
miniconda_path_default()
```

`pymyami_exists` *Check if pymyami is installed*

Description

Check if pymyami is installed

Usage

```
pymyami_exists()
```

Index