Package: volcalc (via r-universe)

November 16, 2026

Title Calculate Volatility of Chemical Compounds

Version 2.2.0.9000

Description Calculate estimated relative volatility index values for organic compounds based on functional group contributions. Calculation uses the SIMPOL.1 method in Prankow and Asher (2008) <doi:10.5194/acp-8-2773-2008> or modified SIMPOL.1 method as in Meredith et al. (2023) <doi:10.5194/acp-8-2773-2008>.

License MIT + file LICENSE

URL https://meredith-lab.github.io/volcalc/, https://cct-datascience.r-universe.dev/volcalc

BugReports https://github.com/Meredith-Lab/volcalc/issues

Depends R (>= 2.10)

Imports ChemmineOB, ChemmineR, dplyr, fs, httr2, magrittr, purrr, rlang, stringr, tibble, tidyr, tidyselect

Suggests knitr, processx, rmarkdown, testthat (>= 3.0.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/pak/sysreqs cmake make libicu-dev libpng-dev librsvg2-dev libuv1-dev libopenbabel-dev libssl-dev

Repository https://cct-datascience.r-universe.dev

Date/Publication 2025-04-24 19:50:43 UTC

RemoteUrl https://github.com/Meredith-Lab/volcalc

RemoteRef HEAD

RemoteSha 98bd95511a96a8af9017ed9b2cc04f46ec094b46

Contents

calc_vol 2
get_fx_groups 4
get_mol_kegg 5
mol_example 6
simpol1 6
smarts_simpol1 8

Index **9**

`calc_vol` *Calculate volatility estimate for a compound*

Description

Estimate relative volatility index value and category for specified compounds using group contribution methods.

Usage

```
calc_vol(
  input,
  from = c("mol_path", "smiles"),
  method = c("meredith", "simpol1"),
  temp_c = 20,
  environment = c("clean", "polluted", "soil"),
  validate = TRUE,
  return_fx_groups = FALSE,
  return_calc_steps = FALSE
)
```

Arguments

`input`	A path to a .mol file or a SMILES string.
`from`	The form of `input`. Either `"mol_path"` (default) or `"smiles"`.
`method`	The method for calculating estimated volatility. See `simpol1()` for more details.
`temp_c`	numeric; For methods that allow it, specify temperature (in degrees C) at which log10(P) estimates are calculated.
`environment`	The environment for calculating relative volatility categories. RVI thresholds for low, moderate, and high volatility are as follows: `"clean"` (clean atmosphere, default) -2, 0, 2; `"polluted"` (polluted atmosphere) 0, 2, 4; `"soil"` 4, 6, 8. For more information about these thresholds see Meredith et al. (2023) and Donahue et al. (2006).
`validate`	logical; if `TRUE` (default), results are checked for possible errors in parsing by Open Babel and `NA`s are returned if possible errors are found. Setting to `FALSE` bypasses these checks—use at your own risk! Validation is not available on Windows. See **Details** of `get_fx_groups()` for more information.

`return_fx_groups`
: When `TRUE`, the returned tibble includes functional group counts.

`return_calc_steps`
: When `TRUE`, the returned tibble includes intermediate volatility calculations. See **Details**.

Details

$\log_{10}C^*$ is used for the calculated relative volatility index (`rvi`). $\log_{10}C^* = \log_{10}(PM/RT)$ where P is the estimated vapor pressure for the compound, M is molecular mass of the compound, R is the universal gas constant, and T is temperature (293.14K or 20ºC). When `return_calc_steps = TRUE`, the log of estimated vapor pressure, `log10_P`, and $\log_{10}(M/RT)$, `log_alpha`, are also returned.

Value

A tibble with relative volatility index (`rvi`) and volatility category (`category`).

References

Donahue, N.M., Robinson, A.L., Stanier, C.O., Pandis, S.N., 2006. Coupled Partitioning, Dilution, and Chemical Aging of Semivolatile Organics. Environ. Sci. Technol. 40, 2635–2643. doi:10.1021/es052297c

Meredith L, Ledford S, Riemer K, Geffre P, Graves K, Honeker L, LeBauer D, Tfaily M, Krechmer J. 2023. Automating methods for estimating metabolite volatility. Frontiers in Microbiology. doi:10.3389/fmicb.2023.1267234

See Also

`get_fx_groups()`, `simpol1()`

Examples

```
mol_paths <- mol_example()
calc_vol(mol_paths)

# Return functional group counts from get_fx_groups()
calc_vol(mol_paths,  return_fx_groups = TRUE)

# Return intermediate calculations
calc_vol(mol_paths, return_calc_steps = TRUE)
```

get_fx_groups *Count compound functional groups*

Description

Returns functional group counts relevant to calculating estimated volatility for specified compounds. Users will not typically interact with this function directly, but rather by using `calc_vol()`.

Usage

```
get_fx_groups(compound_sdf, validate = TRUE)
```

Arguments

`compound_sdf`	a ChemmineR::SDFset object returned by `ChemmineR::read.SDFset()` or `ChemmineR::smiles2sdf()`, for example.
`validate`	logical; if `TRUE` (default), results are checked for possible errors in parsing by Open Babel and `NA`s are returned if possible errors are found. Setting to `FALSE` bypasses these checks—use at your own risk! Validation is not available on Windows. See **Details** for more information.

Details

For more details on how functional groups are defined, see the smarts_simpol1 data set.

It is unfortunately difficult to capture errors and warnings produced by the command line tool Open-Babel used by `ChemmineOB`, a dependency of `volcalc`. These errors and warnings are printed to the R console, but they are *not* R errors and do not stop code from running and producing potentially incorrect data. `validate = TRUE` checks the output of certain OpenBabel procedures for the *symptoms* of these errors, namely missing values for InChI and molecular formula. Unfortunately, since InChI generation is not available with the Windows version of `ChemmineOB`, this validation step cannot be performed on Windows and `validate = TRUE` will simply print a warning that can be silenced by setting `validate = FALSE`.

Value

A tibble with columns of basic compound info and functional group counts.

Note

This function currently does **not** capture the carbon number on the acid-side of amide, one of the functional groups used in SIMPOL.1. Contributions of SMARTS strings or other methods to capture this "functional group" are welcome.

See Also

`calc_vol()`

Examples

```
mol_path <- mol_example()[1]
sdf <- ChemmineR::read.SDFset(mol_path)
get_fx_groups(sdf)
```

`get_mol_kegg` *Download compound .mol files from KEGG*

Description

Downloads mol files corresponding to individual compounds or compounds in a pathway from KEGG.

Usage

```
get_mol_kegg(compound_ids, pathway_ids, dir, force = FALSE)
```

Arguments

`compound_ids`	Character vector of KEGG compound IDs—5 digits prepended with a "C".
`pathway_ids`	Character vector of KEGG pathway or pathway module IDs—5 digits prepended with "map" or "M", respectively.
`dir`	Path to a folder to save .mol files in. Folder will be created if it does not already exist.
`force`	Logical; by default (`FALSE`), .mol files will not be downloaded if they are found in `dir`. Set this to `TRUE` to download and overwrite existing files.

Value

A tibble with the columns `compound_ids`, `pathway_ids` (if used), and `mol_paths` (paths to downloaded .mol files).

Note

For additional functionality for interacting with KEGG, try the `KEGGREST` package, which this function was inspired by.

Examples

```
## Not run:
get_mol_kegg(compound_ids = c("C16181", "C06074"), dir = tempdir())
get_mol_kegg(pathway_ids = "map00253", dir = tempdir())

## End(Not run)
```

mol_example *Example .mol files*

Description

volcalc comes bundled with some example .mol files in its inst/extdata directory. This function provides easy access to them.

Usage

```
mol_example()
```

Details

File names are the KEGG identifiers. Compound names are as follows:

- C00031: D-Glucose
- C00157: Phosphatidylcholine
- C08491: (-)-Jasmonic acid
- C16181: beta-2,3,4,5,6-Pentachlorocyclohexanol
- C16286: Geosmin
- C16521: Isoprene

Value

File paths to installed example .mol files.

Examples

```
#return paths to all example .mol files
mol_example()

#examine the contents of a file
readLines(mol_example()[1])
```

simpol1 *SIMPOL.1 method for calculating estimated volatility*

Description

Implements the SIMPOL.1 group contribution method for predicting liquid vapor pressure of organic compounds as described in Pankow & Asher (2008) and a modified version described in Meredith et al. (2023). Users will not usually use this function directly, but rather through calc_vol().

Usage

```
simpol1(fx_groups, meredith = TRUE, temp_c = 20)
```

Arguments

`fx_groups`	A data.frame or tibble with counts of functional groups produced by `get_fx_groups()` (or manually, with the same column names).
`meredith`	Logical; `FALSE`: use the original SIMPOL.1 method. `TRUE`: use the modified version in Meredith et al. (2023).
`temp_c`	Numeric; the temperature at which to calculate volatility estimates in °C.

Details

The output includes a column for `log10_P` where $\log_{10} P^{\circ}_{\mathrm{L},i}(T) = \sum_k \nu_{k,i} b_k(T)$, or the sum of the counts of functional groups ($\nu_{k,i}$) times the coefficients for each functional group ($b_K(T)$). Units are in log10 atmospheres.

The modified method in Meredith et al. (2023) adds the following additional functional groups:

Using the same coefficient as nitrate

- Phosphoric acid
- Phosphoric ester
- Sulfate
- Sulfonate
- Thiol

Using the same coefficient as ester

- Carbothioester

Value

The `fx_groups` tibble with the additional `log10_P` column.

Note

The method described in Pankow & Asher (2008) was developed using data between 0 °C and 120 °C, so extrapolating beyond that range is not recommended.

References

Meredith L, Ledford S, Riemer K, Geffre P, Graves K, Honeker L, LeBauer D, Tfaily M, Krechmer J. 2023. Automating methods for estimating metabolite volatility. Frontiers in Microbiology. doi:10.3389/fmicb.2023.1267234

Pankow, J.F., Asher, W.E. 2008. SIMPOL.1: a simple group contribution method for predicting vapor pressures and enthalpies of vaporization of multifunctional organic compounds. Atmos. Chem. Phys. doi:10.5194/acp827732008

See Also

calc_vol()

Examples

```
mol_path <- mol_example()[3]
sdf <- ChemmineR::read.SDFset(mol_path)
fx_groups <- get_fx_groups(sdf)
simpol1(fx_groups)
```

---

smarts_simpol1 *Search patterns used for SIMPOL.1 functional groups*

Description

This dataframe documents how functional groups for the SIMPOL.1 and Meredith et al. method are defined using SMARTS strings or ChemmineR functions.

Usage

```
smarts_simpol1
```

Format

method Either "simpol1" for functional groups only used with the SIMPOL.1 method, or "meredith" for additional groups used in the Meredith et al. method.

functional_groups These correspond to matching column names in the results of get_fx_groups().

description Functional group description from Table 5 of Pankow & Asher (2008)

smarts SMARTS strings used to capture groups, when applicable

fun The function used to capture the functional group. When smarts is not NA, this is always "ChemmineR::smartsSearchOB". Other groups are captured with other ChemmineR functions or as calculations using other functional groups.

notes Notes including how any functional group counts are corrected when there is overlap. E.g. when one SMARTS pattern is a subset of another pattern, but the two groups are counted separately without overlap in the SIMPOL.1 method.

References

Meredith L, Ledford S, Riemer K, Geffre P, Graves K, Honeker L, LeBauer D, Tfaily M, Krechmer J. 2023. Automating methods for estimating metabolite volatility. Frontiers in Microbiology. doi:10.3389/fmicb.2023.1267234

Pankow, J.F., Asher, W.E. 2008. SIMPOL.1: a simple group contribution method for predicting vapor pressures and enthalpies of vaporization of multifunctional organic compounds. Atmos. Chem. Phys. doi:10.5194/acp827732008

Index

∗ **datasets**
smarts_simpol1, 8

calc_vol, 2
calc_vol(), *4*, *6*, *8*
ChemmineR::read.SDFset(), *4*
ChemmineR::SDFset, *4*
ChemmineR::smartsSearchOB, *8*
ChemmineR::smiles2sdf(), *4*

get_fx_groups, 4
get_fx_groups(), *2, 3*, *7, 8*
get_mol_kegg, 5

mol_example, 6

simpol1, 6
simpol1(), *2, 3*
smarts_simpol1, *4*, 8