iapws Documentation

Release 0.1.0

M. Skocic

Apr 28, 2023

CONTENTS:

CHAPTER

ONE

GETTING STARTED

1.1 iapw

ipaws is a Fortran library providing the formulas for computing light and heavy water properties. It also provides a API for the C language. The formulations are taken from http://iapws.org. A shared and a static library *libiapws* are compiled (f2008+) with the Fortran and C headers. The static and shared libraries can be installed in order to be included in Fortran or C programs.

The compilation was tested on Linux (Debian), MacOS and Windows.

1.1.1 How to install

Create build directory

```
$ mkdir build
$ d build
```

Generate a makefile

- On Unix-like OS:

```
cmake -G "Unix Makefiles" -S .. -DCMAKE_BUILD_TYPE=release -DCMAKE_INSTALL_PREFIX=/
↪path/to/folder
```

- On windows with MSYS2:

```
cmake -G "Unix Makefiles" -S .. -DCMAKE_BUILD_TYPE=release -DCMAKE_INSTALL_PREFIX=/
↪path/to/folder
```

- On windows with ifort and msvc:

```
cmake -G "NMake Makefiles" -S .. -DCMAKE_BUILD_TYPE=release -DCMAKE_INSTALL_PREFIX=/
↪path/to/folder
```

Build

```
cmake --build .
```

Run tests

```
ctest
```

Install

```
cmake --install .
```

1.1.2 Dependencies

```
gcc>=4.6 or msvc>=14

gfortran>=4.6 or ifort>=18

cmake>=3.10
```

1.1.3 License

GNU General Public License v3 (GPLv3)

1.2 pyiapws

Python wrapper around the Fortran iapws library. Follow the installation instructions. for compiling and installing the library.

For now,the wrapper must be compiled after installing the iapws library using the compiler that was used to compile your python interpreter.

1.2.1 How to install

```
pip install pyiapws
```

1.2.2 Dependencies

```
numpy>=1.20
```

1.2.3 License

GNU General Public License v3 (GPLv3)

1.3 Examples

1.3.1 Example in Fortran

```
program example_in_f
    use iso_fortran_env
    use iapws
    implicit none
    real(real64) :: kh, kd
    character(len=5) :: gas = "O2"
    character(len=5) :: solvent = "H2O"
    real(real64) :: T = 25.0d0

    kh = iapws_kh(T, gas, solvent)
    print "(A10, 1X, A10, 1X, A2, F10.1, A, 4X, A3, SP, F10.4)", "Gas=", gas, "T=", T,
↪ "C", "kh=", kh
```

(continues on next page)

(continued from previous page)

```
    kd = iapws_kd(T, gas, solvent)
    print "(A10, 1X, A10, 1X, A2, F10.1, A, 4X, A3, SP, F15.4)", "Gas=", gas, "T=", T,
↪ "C", "kh=", kd

end program
```

1.3.2 Example in C

```c
#include <string.h>
#include <stdio.h>
#include "iapws.h"

int main(int argc, char **argv){

    double T = 25.0; /* in C*/
    char *gas = "O2";
    char *solvent = "H2O";
    double kh, kd;

    if(argc > 1 ){
        printf("%s\n", argv[1]);
    }

    kh = iapws_capi_kh(T, gas, solvent, strlen(gas), strlen(solvent));
    printf("Gas=%s\tT=%fC\tkh=%+10.4f\n", gas, T, kh);

    kd = iapws_capi_kd(T, gas, solvent, strlen(gas), strlen(solvent));
    printf("Gas=%s\tT=%fC\tkd=%+15.4f\n", gas, T, kd);

    return 0;
}
```

CHAPTER

TWO

IAPWS - THEORETICAL BACKGROUND

2.1 IAPWS G7-04

The computation is based on the parameters provided by the IAPWS 2004 [1].

2.1.1 Henry Contant: kh

The Henry constant kH is defined as shown in equation Eq.2.1.1.

$$k_H = \lim_{x_2 \to 0} f_2/x_2 \tag{2.1.1}$$

- f_2: liquid-phase fugacity
- x_2: mole fraction of the solute

The Henry's constant k_H is given as a function of temperature by:

$$\ln\left(\frac{k_H}{p_1^*}\right) = A/T_R + \frac{B \cdot \tau^{0.355}}{T_R} + C \cdot T_R^{-0.41} \cdot \exp\tau \tag{2.1.2}$$

- $\tau = 1 - T_R$
- $T_R = T/T_{c1}$
- T_{c1}: critical temperature of the solvent as recommended by IAPWS [2] (647.096 for H2O and 643.847 K for D2O)
- p_1^* is the vapor pressure of the solvent at the temperature of interest and is calculated from the correlation of Wagner and Pruss for H2O [3] and from the correlation of Harvey and Lemmon for D2O [4].

Both equations have the form:

$$\ln\left(p_1^*/p_{c1}\right) = T_R^{-1} \sum_{i=1}^{n} a_i \tau^{b_i} \tag{2.1.3}$$

- n is 6 for H2O and 5 for D2O
- p_{c1} is the critical pressure of the solvent recommended by IAPWS [2] (22.064 MPa for H2O and 21.671 MPa for D2O)

2.1.2 Vapor-Liquid Distribution Constant: kd

The liquid-vapor distribution constant kD is defined as shown in equation Eq.2.1.4.

$$k_D = \lim_{x_2 \to 0} y_2/x_2 \tag{2.1.4}$$

- x_2: mole fraction of the solute
- y_2 is the vapor-phase solute mole fraction in equilibrium with the liquid

The vapor-liquid distribution constant *k_D* is given as a function of temperature by:

$$\ln K_D = qF + f(\tau) + (F + G\tau^{2/3} + H\tau) \exp\left(\frac{273.15 - T(K)}{100}\right) \tag{2.1.5}$$

- q : -0.023767 for H2O and -0.024552 for D2O.
- $f(\tau)$ [3] for H2O and [5] for D2O.

In both cases, $f(\tau)$ has the following form:

$$f(\tau) = \sum_{i=1}^{n} c_i \cdot \tau^{d_i} \tag{2.1.6}$$

- n is 6 for H2O and 4 for D2O

2.1.3 Molar fractions

$$\begin{aligned} x_2 &= \frac{1}{k_H} \\ y_2 &= \frac{k_D}{k_H} \end{aligned} \tag{2.1.7}$$

CHAPTER

THREE

RELEASE NOTES

3.1 iapws 0.1.0 Release Note

3.1.1 Changes

- Implementation of kH and kD from IAPWS G7-04 in fortran + C API
- Python wrapper for kH and kD.

3.1.2 Download

iapws

pyiapws

3.1.3 Contributors

Milan Skocic

3.1.4 Commits

Full Changelog: https://github.com/MilanSkocic/pyiapws/compare/….0.1.0

CHAPTER

FOUR

AUTOGENERATED DOCUMENTATION

4.1 iapws

namespace **iapws**

Main module for IAPWS computations.

Functions

pure real(real64) function, public iapws_kh (t, gas, solvent)

Compute the henry constant for a given temperature and gas in solvent.

Parameters

- **T** – **[in]** Temperature in °C.
- **gas** – **[in]** Gas.
- **solvent** – **[in]** Solvents: H2O or D2O. Default is H2O.

Returns

kh Henry constant in mole fraction per GPa. NaN if gas not found.

pure real(real64) function, public iapws_kd (t, gas, solvent)

Compute the vapor-liquid constant for a given temperature and gas in solvent.

Parameters

- **T** – **[in]** Temperature in °C.
- **gas** – **[in]** Gas.
- **solvent** – **[in]** Solvents: H2O or D2O. Default is H2O.

Returns

kd Vapor-liquid constant. NaN if gas not found.

namespace **iapws_capi**

C API for the IAPWS module.

Functions

real(c_double) function, public iapws_capi_kh (t, gas, solvent, size_gas, size_solvent)

Compute the henry constant for a given temperature and gas in solvent.

Parameters

- **T** – **[in]** Temperature in °C.
- **gas** – **[in]** Gas.
- **solvent** – **[in]** Solvents: H2O or D2O. Default is H2O.
- **size_gas** – **[in]** Length of the string gas.
- **size_solvent** – **[in]** Length of the string gas.

Returns

kh Henry constant. NaN if gas not found.

real(c_double) function, public iapws_capi_kd (t, gas, solvent, size_gas, size_solvent)

Compute the vapor-liquid constant for a given temperature and gas in solvent.

Parameters

- **T** – **[in]** Temperature in °C.
- **gas** – **[in]** Gas.
- **solvent** – **[in]** Solvents: H2O or D2O. Default is H2O.
- **size_gas** – **[in]** Length of the string gas.
- **size_solvent** – **[in]** Length of the string gas.

Returns

kd Vapor-Liquid constant. NaN if gas not found.

4.2 pyipaws

IAPWS computations.

pyiapws.iapws.**kd**(*temperature: int | float | ndarray[Any, dtype[ScalarType]]*, *gas: str*, *solvent: str*) → float | ndarray[Any, dtype[ScalarType]]

Compute the vapor-liquid distribution constant for the gas and solvent at temperature.

Parameters

temperature: int, float or array-like.
Temperature in °C.

gas: str
Desired gas.

solvent: str
Desired solvent: H2O or D2O.

Returns

kd: float or array-like
Henry constant.

pyiapws.iapws.**kh**(*temperature: int | float | ndarray[Any, dtype[ScalarType]]*, *gas: str*, *solvent: str*) → float | ndarray[Any, dtype[ScalarType]]

Compute the Henry constant for the gas and solvent at temperature.

Parameters

temperature: int, float or array-like.
Temperature in °C.

gas: str
Desired gas.

solvent: str
Desired solvent: H2O or D2O.

Returns

kh: float or array-like
Henry constant.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

BIBLIOGRAPHY

[1] IAPWS. Guideline on the Henry's Constant and Vapor-Liquid Distribution Constant for Gases in H\textsubscript 2O and D\textsubscript 2O at High Temperatures. Technical Report G7-04, IAPWS, Kyoto, Japan, 2004.

[2] IAPWS. Revised Release on the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam. Technical Report R7-97, IAPWS, Lucerne, Switzerland, 2007.

[3] Wolfgang Wagner and A. Pruss. International Equations for the Saturation Properties of Ordinary Water Substance. Revised According to the International Temperature Scale of 1990. Addendum to J. Phys. Chem. Ref. Data 16, 893 (1987). *Journal of Physical and Chemical Reference Data*, 22(3):783–787, May 1993. doi:10.1063/1.555926.

[4] Allan H. Harvey and Eric W. Lemmon. Correlation for the Vapor Pressure of Heavy Water From the Triple Point to the Critical Point. *Journal of Physical and Chemical Reference Data*, 31(1):173–181, March 2002. doi:10.1063/1.1430231.

[5] R. Fernandez-Prini, J.L. Alvarez, and A.H. Harvey. Henry's Constants and Vapor–Liquid Distribution Constants for Gaseous Solutes in H2O and D2O at High Temperatures. *Journal of Physical Chemistry Reference Data*, 32(2):903–916, 2003.

PYTHON MODULE INDEX

p

INDEX