# EasyProp_MATLAB_Interface.m

## Table of Contents



```
This script will outline the functionality availabe for the EasyProp
interface from a MATLAB environment.  If there is additional
functionality that you believe would be helpful, please let me know.
```

# Prepare the Environment

```
clear
clc
close 'all'

EasyProp_path='..'; % <-- enter relative or absolute path to folder containing
 EasyProp.py
if count(py.sys.path,EasyProp_path) == 0  % <-- see if desired directory is on
 path
    insert(py.sys.path,int32(0),EasyProp_path); %<-- if not; add it.
end
```

# Initialize the fluid:

```
fluid = 'Water';
```

```
Some other available fluids include:
```

1. 'Water'

2. 'Air'

3. 'CO2'
4. 'He'
5. 'N2'
6. 'Xe'
7. ...and much, much, more...
8. complete list available at: http://www.coolprop.org/fluid_properties/PurePseudoPure.html#list-of-fluids

```
units = 'SI'; % other choice is: 'USCS'

fluid = py.EasyProp.simpleFluid(fluid,units);
```

# Enthalpy - SI units: kJ/kg USCS units: BTU/lbm

```
h1 = fluid.hL_p(101.3); % saturated liquid enthalpy at 101.3 kPa
h2 = fluid.hV_p(8000.0); % saturated vapor enthalpy at 8000 kPa
h3 = fluid.h_ps(3500,1.25); % enthalpy at 3500 kPa, 1.25 kJ/kg-K
h4 = fluid.h_pT(101.3,200); % enthalpy at 101.3 kPa, 200 C
h5 = fluid.h_Tx(350,0.5); % enthalpy at 350 C, 50% quality
h6 = fluid.h_px(8000,0.25); % enthalpy at 8000 kPa, 25% quality
% also: h_pv(P,V)
```

# Entropy - SI units: kJ/kg-K USCS units: BTU/lbm-R

```
s1 = fluid.sL_p(101.3); % saturated liquid entropy at 101.3 kPa
s2 = fluid.sV_p(8000.0); % saturated vapor entropy at 8000 kPa
s3 = fluid.s_Tx(200,0.4); % saturated mixture entropy at 200 C, x=0.4
s4 = fluid.s_pT(101.3,200); % entropy at 101.3 kPa, 2000 C
s5 = fluid.s_ph(300.0,1234); % entropy at 300 kPa, h=1234 kJ/kg
s6 = fluid.s_px(300, 0.5); % entropy at 300 kPa, 50% quality
% also s_pu(P,U)
```

# Temperature - SI units: C USCS units: F

```
T1 = fluid.T_ps(101.3,0.5); % temperature at 101.3 kPa, 0.5 kJ/kg-K
T2 = fluid.T_ph(101.3,1234); % temperature at 101.3 kPa, 1234 kJ/kg
T3 = fluid.T_pv(200.0,0.002); % temperature at 200 kPa, 0.002 m^3/kg
T4 = fluid.Tsat_p(8000); % saturation temperature at 8000 kPa
T5 = fluid.T_crit(); % critical temperature for the fluid (C for SI, F for
 USCS)
T6 = fluid.T_pu(101.3,200.); % temperature at 101.3 kPa, 200 kJ/kg
T7 = fluid.T_sv(1.5,3.5); % temperature at 1.5 kJ/kg-K and 3.5 m^3/kg
```

# Pressure - SI units: kPa USCS units: psia

```
P1 = fluid.Psat_T(300); % saturation pressure at 300 C
P2 = fluid.P_vT(0.002,300.); % pressure at 0.002 m^3/kg, 300 C
```

```
P3 = fluid.P_crit(); % critical pressure for the fluid (kPa or psia)
% also: P_sT(S,T)
%       P_sh(S,H)
```

# Internal energy - SI units: kJ/kg, USCS units: BTU/lbm

```
U1 = fluid.uL_p(101.3); % internal energy of saturated liquid at 101.3 kPa
U2 = fluid.uV_p(101.3); % internal energy of saturated vapor at 101.3 kPa
U3 = fluid.u_px(101.3,0.5); % internal energy of saturated mixture at 101.3
 kPa, 50% quality
U4 = fluid.u_Tx(200,0.5); % internal energy of saturated mixture at 200 C, 50
 % quality
U5 = fluid.u_pT(101.3,200); % internal energy of a fluid at 101.3 kPa, 200 C
U6 = fluid.u_ps(101.3,1.5); % internal energy of a fluid at 101.3 kPa, 1.5 kJ/
kg-K
U7 = fluid.u_Tv(20,0.002); % internal energy of a fluid at 20 C, 0.002 m^3/kg
```

# specific volume - SI units: m^3/kg, USCS units: ft^3/lbm

```
V1 = fluid.v_ph(101.3,2300.); % specific volume of fluid at 101.3 kPa, 2300.0
 kJ/kg
V2 = fluid.v_pu(101.3,500.); % specific volume of fluid at 101.3 kPa, 500. kJ/
kg
V3 = fluid.v_Tx(200.,0.5); % specific volume for saturated mixture at 200 C,
 50% quality
V4 = fluid.v_pT(101.3,200.); % specific volume for 101.3 kPa, 200 C
```

# Quality

```
x1 = fluid.x_ph(100,700); % quality at 100 kPa, 700 kJ/kg enthalpy
x2 = fluid.x_pu(101,500); % quality at 101 kPa, 500 kJ/kg internal energy
```

# viscosity - SI units: Pa-s (kg/m-s) USCS units: lbm/ft-s

```
mu1 = fluid.mu_pT(101.3,200); % viscoisty at 101.3 kPa, 200 C
```

# thermal conductivity - SI units: kW/m-K USCS units: BTU/ft-sec-R

```
k1 = fluid.k_pT(101.3,200); % thermal conductivity at 101.3 kPa, 200 C
```

# Critical point properties

```
Pcrit = fluid.P_crit(); % critical pressure
```

```
Tcrit = fluid.T_crit(); % critical temperature
```

# Prandtl Number

```
Pr1 = fluid.Prandtl_pT(101.3,200); % fluid Prandtl number at 101.3 kPa, 200 C

helium = py.EasyProp.simpleFluid('He','USCS');
```

# Gas Constant - SI units kJ/kg-K USCS units: BTU/lbm-R

```
R1 = helium.R_pT(14.7,200); % Gas constant for Helium at 14.7 psia, 200 F
```

# Molecular weight - SI kG/kmol USCS units: lbm/mol

```
M = helium.M();
```

# Specific heats - SI units kJ/kg-K, USCS units: BTU/lbm-R

```
Cp = helium.Cp_pT(14.7,200); % specific heat at constant pressure for Helium
 at 14.7 psia, 200 F
Cv = helium.Cv_pT(14.7,200); % specific heat at constant volume
gamma = helium.gamma_pT(14.7,200); % ratio of: Cp/Cv for Helium
```

# Humid Air Object and functions

```
humidAir = py.EasyProp.HumidAir('SI'); % option: 'SI' | 'USCS'

% humidity ratio as a function of temp, pressure, and relative humidity
w = humidAir.w_PTR(101.3,20,0.2); % no units

% enthalpy as a function of temp, pressure, and relative humidity
h = humidAir.h_PTR(101.3,20,0.2);% kJ/kg or BTU/lbm
```

# Simple Mixtures

example mixture of 35 w/o Helium, 65 w/o CO2

```
fluidMix = py.dict(pyargs('He',0.35,'CO2',0.65));
myMix = py.EasyProp.simpleMixture(fluidMix,'w/o','SI');

% -- (mixtures are under construction; stay away!!)
```

*Published with MATLAB® R2022a*