NIST RefProp Add-in
Reference Fluid Thermodynamic and Transport Properties

RefProp Mathcad Add-in: Version 2.0

Mathcad wrapper functions for
NIST Reference Fluid Thermodynamic and Transport Properties Database,
Version 9.1 and higher

NIST RefProp DLL Currently Loaded

NIST RefProp Library: Version 10.0.0.0

Release Date: 14/11/2020

Contents

About the NIST RefProp Mathcad Add-in

Introduction

This User's Guide accompanies the **NIST RefProp Mathcad Add-in.** The add-in contains over 50 functions that return thermodynamic and transport properties computed from the **NIST Standard Reference Database**. To make the NIST RefProp DLL available to this add-in, NIST RefProp 9.1.1 or later must be installed on the machine. This User's Guide contains live math documents and text, explaining how all of these functions work, and using them to calculate real-world examples. It also has an index of all functions contained in the Add-in.

Once the Add-in is installed, these functions can be used in any Mathcad document, either though the Insert Function dialog or using the provided reference worksheet with units. They produce output that can be used in the same way as output from the operators and functions that are part of Mathcad.

Organization

This Guide is organized into seven chapters and five appendices that fully explain the add-in interface to the **RefProp** Functions and give examples of their usage. A Reference Section and Function Index are also included.

Units

All of the **NIST RefProp** functions adhere to the SI system of units, excepting pressures in [kPa] and mass based energy units of [kJ/kg]. The **RefProp Mathcad Add-in** functions are mass based only and follow the same convention, except that pressure values are passed and retuned in units of [MPa]; mostly to keep numerical input/output values manageable and of an order of magnitude that minimizes roundoff error when working with these numbers in Mathcad. Conversion between [MPa] and [kPa] is handled by the wrapper code. This is consistent with unit standards of the IAPWS and ASME functions and databases. Although the parameters passed to the direct functions should be converted to the proper units, they should not have actual Mathcad units attached.

Throughout this Guide, examples are given for stripping and re-applying units for each of the **RefProp Add-in** functions.

Double Precision

The **NIST RefProp** functions are implemented in double precision Fortran and the **RefProp Mathcad Add-in** wrapper functions are implemented in double precision C++.

Chapter 1 Introduction

About RefProp

REFPROP is an acronym for **REF**erence fluid **PROP**erties. This program, developed by the National Institute of Standards and Technology (NIST) provides formulations of the thermodynamic and transport properties of industrially important fluids and their mixtures with an emphasis on refrigerants and hydrocarbons. These properties can be displayed in tables and plots through the graphical user interface; they are also accessible through spreadsheets or user-written applications accessing the REFPROP DLL or the FORTAN property subroutines.

About the Mathcad Add-In

The REFPROP formulations are coded in standard Fortran-77 and compiled into a Dynamically Linked Library (DLL). This library of routines is loaded by a Mathcad wrapper function library, created in Microsoft Visual C++, and compiled into it's own DLL. Each wrapper function calls its corresponding NIST RefProp function in REFPROP.DLL and passes parameters and results to and from the Mathcad interface.

The Mathcad functions have been "verified" to give the same answers as can be obtained by other implemented wrappers of the REFPROP database to a precision 6 significant figures or better. "Validation" of the REFPROP database for accuracy should be obtained independently by users of this code for the specific materials of interest.

Version 1.0 of this Mathcad Add-In provided functionality for ***pure fluids only***. Support for ***mixtures*** of pure components was not supported due to inaccuracies observed in the mixture models. Version 1.0 was based on the library files from, version 7.1 of the NIST REFPROP Database, which were linked into the Mathcad wrapper DLL directly as a LIB file.

Version 1.1 added some additional reverse function calls, and incorporated the library files from version 8.0 of the NIST REFPROP database.

Version 2.0 is a rewrite of the wrapper function stubs to load the NIST REFPROP.DLL from the user's machine. The NIST RefProp software must be installed (purchased from NIST), in addition to the RefProp Mathcad Add-in DLL. New features in version 2.0 include

- Loading of predefined mixture files in addition to pure and pseudo-pure fluids
- New fluids and mixtures available in RefProp 9.1 and later,
- More flexible and automatic location of the fluid files on the user's machine,
- Specification of custom mixtures in the fluid string,
- Dynamic use of the currently installed RefProp version DLL,
- Static linking of the Intel Fortran runtime libraries with RefProp 9.1.1, removing the Intel Composer (Fortran) Run-Time Library as an additional installation requirement.

Chapter 2 RefProp Add-in Installation

Prerequisites

The latest NIST RefProp program ***must*** be installed on the machine or the Mathcad Add-in will not run (user will be notified if it is not and the Add-in will not load). This program is available for purchase from NIST. It is recommended that NIST RefProp 9.1.1 or later being installed as this version greatly improves installation and integration issues with other codes.

NOTE: Earlier versions of NIST RefProp may work, however, they will most likely require the installation of the Intel Composer (Fortran) Redistributable Runtime Components as previous REFPROP.DLL and REFPROP64.DLL libraries were dynamically linked with these libraries. Version 9.1.1 and later DLLs link to these Fortran components statically, so they are included within the DLL and not required on the machine.

Installing the NIST RefProp Add-in

Compiling and Installing on Mathcad Prime

The procedure for compiling and installing the NIST RefProp add-in for Mathcad Prime is identical to the instructions above, with a few minor differences:

1. The Visual Studio 2015 solution file is located in the **\buildPrime** directory of the code repository.

2. Ensure that the project configuration is set to Release | x64, as this is where file path configuration is stored and builds a 64-bit DLL for Mathcad Prime.

3. The PrimeREFPROPwrapper.DLL is copied into the Mathcad Prime **\Custom Functions** directory. This is typically located here:

   ```
   C:\Program Files\PTC\Mathcad Prime 6.0.0.0\Custom Functions
   ```

   When compiling the wrapper DLL using the provided Visual Studio 2015 solution, this copy procedure is performed automatically.

The above path should be modified in the project properties if not using Mathcad Prime version 6.0.0.0 to reflect the correct location of the user's Mathcad installation in the following locations:

- C/C++ | General > Additional Include Directories
- Linker | Input > Additional Dependencies
- Build Events | Post-Build Event > Command Line & Description

NOTE: Mathcad Prime must be shut down when compiling the RefProp add-in or the files will not be copied to the Mathcad Prime installation directory.

Installation on Mathcad 15

The NIST RefProp add-in for Mathcad is very simply installed by copying the compiled MathcadREFPROPwrapper.DLL into the Mathcad 15 **userefi** directory. This is typically located here:

```
C:\Program Files (x86)\Mathcad\Mathcad 15\userefi
```

When compiling the wrapper DLL using the provided Visual Studio 2015 solution, the resulting target DLL is copied automatically as a Post-Build Event, provided that the DLL compiled correctly and the users has write access to the **userefi** directory.

Compiling the Mathcad 15 DLL

Go to the /build15 directory of the wrapper repository and open the Visual Studio 2015 solution in Visual studio. Make sure that the project configuration is set to "**Release**" (not Debug) for "**x86**". This compiles the 32-bit version of the DLL add-in for Mathcad 15. Then select **Build | Build MathcadREFPROPwrapper** from the main menu. The DLL and support files should be copied to the Mathcad 15 installation directory automatically (user must have write access to Mathcad's **userefi** folder).

NOTE: Mathcad 15 must be shut down when compiling the RefProp add-in or the files will not be copied to the Mathcad 15 installation directory.

Help Files

The ***NIST RefProp*** Handbook, context sensitive help, and Insert Function assistant will all be copied to the appropriate Mathcad 15 installation directories by the Pre-Build Event in the Visual Studio solution. These files can can also be copied to the appropriate directories manually if not building the DLL with Visual Studio.

Alternate NIST RefProp Location (Optional)

The RefProp DLLs as well as the fluids and mixtures files from NIST are stored and accessed from the RefProp installation directory on the user's hard drive. If NIST RefProp has been installed in a non-default location, the add-in does a pretty good job of finding it. However, you can manually point the Mathcad DLL to that location with a user environment variable, **NIST_PATH**.

The location provided should have two subdirectories underneath it,

{*NIST_PATH*}
\fluids
\mixtures

Fluid and mixture files should be loaded in these subdirectories appropriately.

Uninstalling the NIST RefProp Add-in

The NIST RefProp Add-in can be uninstalled very easily by removing the add-in DLL from the Mathcad 15\userefi directory or the Mathcad Prime \Custom Functions directory. They can be moved temporarily to a \save subdirectory to keep them from loading with Mathcad 15 or Prime is launched.

Chapter 3 General Usage

The RefProp Add-in Functions

Once ***RefProp*** is installed, the ***RefProp*** functions will be registered by Mathcad Prime. Unfortunately, Mathcad Prime (as of version 6.0) does not yet have a facility to integrate these functions into the user interface. However, these functions follow a specific naming convention. This is *exactly* the naming convention used in other Mathcad add-ins such as the legacy **ASME Steam Tables add-in** from Adept Science (*no longer distributed or supported*) and the **IF97 add-in** from CoolProp.

- Each function starts with the prefix "***rp_***". This avoids conflicts with any other add-ins that may be loaded.

- The next characters in the function identify the physical property to be calculated. The available properties are:

  Thermodynamic properties
  rho (density),
  h (enthalpy),
  u (internal energy),
  s (entropy),
  cp (isobaric specific heat),
  cv (isochoric specific heat),
  w (speed of sound)

  Transport properties
  mu (viscosity)
  k (thermal conductivity)

  State Point
  t (temperature)
  p (pressure)
  tsat (saturation temperature)
  psat (saturation pressure)

- The property may be followed by either f (liquid) or g (gas) to get the ***saturation*** state, typically as a function of either the saturation temperature or saturation pressure (e.g. ***hf*** = saturated liquid/fluid enthalpy, ***sg*** = saturated vapor/gas entropy).

Fluid Material Selection

Fluid properties for each material in the ***RefProp*** database are stored in individual files `fluids\` and `mixtures\` sub-folders of the ***RefProp*** installation directory.

Each of the property functions requires a fluid string as it's first parameter. This parameter communicates to the ***RefProp*** internal routines which fluid material file should be used to calculate the material properties.

> *NOTE: Each fluid property function makes a call internally to the RefProp* ***SETUP*** *routine, which loads the fluid file from the database location. However, if the fluid requested is already loaded, no action is taken; eliminating unnecessary file I/O.*

Details:

- The "**fluid string**" for a **Pure Fluid** file to be loaded from ***RefProp*** can (optionally) end with an extension of "**.fld**". If the suffix is omitted, RefProp will assume an extension of "**.fld**" and look for the fluid in the "`fluids\`" sub-folder.

  e.g. to load the properties for HELIUM, define a variable such as:

  fluid := "HELIUM.fld" *or* *fluid* := "HELIUM"

- **Pseudo-Pure fluids** are mixtures that have been re-fit to load like a pure fluid. The "**fluid string**" for these files ***must*** end with the extension "**.ppf**".

  e.g. to load the pseudo-pure properties for AIR, define a variable such as:

  fluid := "AIR.ppf"

- Predefined Mixtures are located in the "`mixtures\`" sub-folder. The "**fluid string**" must use the suffix "**.mix**".

  e.g. to load the mixture properties for AIR, define a variable such as:

  fluid := "AIR.mix"

- If the suffix is omitted, RefProp will look preferentially in the "`fluids\`" folder for a **.fld** file, then a **.ppf** file, and then in the "`mixtures\`" folder for a **.mix** file.

- An **Ad-hoc Mixture** can also be specified in the "**fluid string**" as outlined in **Chapter 7 - Mixture Properties**.

- Available pure fluid files as of ***RefProp 9.1*** are shown below in **Table 1**, pseudo-pure fluids in **Table 2**, and mixture files are listed in **Table 3**. However, users should consult their latest version documentation.

Table 1 : RefProp Pure Fluid Files

Pure Fluids use extension ".fld" (optional)			
acetone	hexane	oxylene	r1233zd
ammonia	hydrogen	octane	r1234yf
argon	hcl	orthohyd	r1234ze
benzene	h2s	oxygen	r124
butane	isobutan	pxylene	r125
1butene	ibutene	parahyd	r13
co2	ihexane	pentane	r134a
co	ioctane	c4f10	r14
cos	ipentane	c5f12	r141b
c2butene	krypton	propane	r142b
cyclohex	mxylene	c3cc6	r143a
cyclopen	md2m	propylen	r152a
cyclopro	md3m	propyne	r161
d4	md4m	so2	r21
d5	mdm	sf6	r218
d6	methane	toluene	r22
decane	methanol	t2butene	r227ea
d2	mlinolea	cf3i	r23
dee	mlinolen	c11	r236ea
dmc	moleate	water	r236fa
dme	mpalmita	xenon	r245ca
c12	mstearat	novec649	r245fa
ethane	c1cc6	r11	r32
ethanol	mm	r113	r365mfc
ebenzene	neon	r114	r40
ethylene	neopentn	r115	r41
fluorine	nitrogen	r116	rc318
d2o	nf3	r12	re143a
helium	n2o	r1216	re245cb2
heptane	nonane	r123	re245fa2
			re347mcc

Table 2 : RefProp Pseudo-Pure Fluid Files

Pseudo-Pure Fluids use extension ".ppf"				
Air	R404A	R407A	R410A	R507A

Table 3 : RefProp Predefined Mixture Files

Pre-defined Mixtures use extension ".mix"				
R401A	R408A	R419A	R431A	R503
R401B	R409A	R420A	R432A	R504
R401C	R409B	R421A	R433A	R507A
R402A	R410A	R421B	R434A	R508A
R402B	R410B	R422A	R435A	R508B
R403A	R411A	R422B	R436A	R509A
R403B	R411B	R422C	R436B	R510A
R404A	R412A	R422D	R437A	R512A
R405A	R413A	R423A	R438A	AIR
R406A	R414A	R424A	R441A	AMARILLO
R407A	R414B	R425A	R442A	EKOFISK
R407B	R415A	R426A	R443A	GLFCOAST
R407C	R415B	R427A	R444A	HIGHCO2
R407D	R416A	R428A	R500	HIGHN2
R407E	R417A	R429A	R501	
R407F	R418A	R430A	R502	

Chapter 4 Applying Units to RefProp Functions

Creating User Function Wrappers

Parameters to the ***RefProp*** functions must be supplied in the required units (as specified in the help text for each function under Insert Function). However, these parameters cannot have actual Mathcad Units attached to them (i.e. they should be unit-less numbers); otherwise incorrect values may be returned as a function result. The results, too, will be returned in the specified units.

To enable usage of Mathcad units and handle unit conversions automatically, create a Mathcad user function that:

1. Divides all of the parameters by the units required by the ***RefProp*** function.
2. Multiplies the result by the units of the ***RefProp*** function return value.

Example:

The RefProp function **rp_hfp**(*fluidstr, p*) returns the saturated liquid enthalpy in [***kJ/kg***]. The parameter, p, is the input saturation pressure [***MPa***].

Create a user function to handle the units according to the instructions above:

$$h_{lsat}(fluidstr, p) := \text{rp\_hfp}\left(fluidstr, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg}$$

Note: In Mathcad, kJ is not a pre-defined unit, so it is defined in the units reference worksheet (see below).

We can now use the wrapper function, instead of calling the ***RefProp*** function directly.

$fluid := \text{“WATER.fld”}$ *Specify Water as the fluid*

$P := 1200 \cdot psi$ *Pressure can then be supplied in any pressure units.*

$h_{lsat}(fluid, P) = 571.938 \ \frac{BTU}{lb}$ *The result can be displayed in any specific energy units.*

Using the Supplied Reference Sheet

Creating function wrappers for all of the functions required can be tedious and provide for inconsistencies between worksheets. A complete set of function wrappers has been supplied in the reference worksheet, **`Refprop_units.xmcd`**, in the ***Units*** directory (see Appendix C).

The **`Refprop_units.xmcd`** worksheet can be ***referenced*** near the top of any worksheet in which you want to use the wrapper functions.

To reference the units worksheet:

1. Choose **Input/Output** from the main menu.
2. Select the Include Worksheet button on the ribbon.

The reference will show up in the worksheet at the cursor location as shown:

Click on the [Include <<] button and browse to the local or shared location of the ***RefProp_units.mcdx*** file. The reference inserted can be edited to make it a relative reference to the local worksheet, or left as an absolute reference to a shared location for permanent/shared access.

All of the pre-defined wrapper functions in the reference worksheet can now be used:

$Name(fluid,1) = \text{“Water”}$ $\qquad$ $CASN(fluid,1) = \text{“7732-18-5”}$

$T_c(fluid,0) = 705.103\ °F$ $\qquad$ $P_c(fluid,0) = 3200\ psi$

$T_2 := 0.5 \cdot T_c(fluid,0)$ $\qquad$ $\rho_{tp}(fluid, T_2, P_c(fluid,0)) = 62.254\ \frac{lb}{ft^3}$

$h_{tp}(fluid, T_2, P_c(fluid,0)) = 98.849\ \frac{BTU}{lb}$

$h_{ft}(fluid, T_2) = 90.716\ \frac{BTU}{lb}$ $\qquad$ $h_{gt}(fluid, T_2) = 1114.4\ \frac{BTU}{lb}$

The available functions in the reference worksheet are listed below. Keep in mind that each function requires, as its first parameter, a "**fluid string**".

RefProp Property Notation

- Fluid Properties
  - ρ Density
  - h Enthalpy
  - u Internal energy
  - s Entropy
  - C_p Isobaric Specific Heat
  - C_v Isochoric Specific Heat
  - μ Kinematic Viscosity
  - ν Dynamic Viscosity (m/r)
  - k Thermal Conductivity
  - Pr Prandtl Number
- Subscript Notation for thermodynamic state:
  - tp Sub-cooled liquid or Super-heated vapor as a function of temperature (t) and pressure (p)
  - ft Saturated Liquid as a function of temp. (t)
  - fp Saturated Liquid as a function of pressure (p)
  - gt Saturated vapor as a function of temp. (t)
  - gp Saturated vapor as a function of pressure (p)

Property Functions

Property	Sub-cooled liquid / super-heated vapor	Saturated Liquid	Saturated Vapor
Density	$\rho_{tp}(fl,T,P)$	$\rho_{fp}(fl,P)$, $\rho_{ft}(fl,T)$	$\rho_{gp}(fl,P)$, $\rho_{gt}(fl,T)$
Enthalpy	$h_{tp}(fl,T,P)$	$h_{fp}(fl,P)$, $h_{ft}(fl,T)$	$h_{gp}(fl,P)$, $h_{gt}(fl,T)$
Internal Energy	$u_{tp}(fl,T,P)$	$u_{fp}(fl,P)$, $u_{ft}(fl,T)$	$u_{gp}(fl,P)$, $u_{gt}(fl,T)$
Entropy	$s_{tp}(fl,T,P)$	$s_{fp}(fl,P)$, $s_{ft}(fl,T)$	$s_{gp}(fl,P)$, $s_{gt}(fl,T)$
Isobaric Specific Heat	$C_{ptp}(fl,T,P)$	$C_{pfp}(fl,P)$, $C_{pft}(fl,T)$	$C_{pgp}(fl,P)$, $C_{pgt}(fl,T)$
Isochoric Specific Heat	$C_{vtp}(fl,T,P)$	$C_{vfp}(fl,P)$, $C_{vft}(fl,T)$	$C_{pgp}(fl,P)$, $C_{pgt}(fl,T)$
Kinematic Viscosity	$\mu_{tp}(fl,T,P)$	$\mu_{fp}(fl,P)$, $\mu_{ft}(fl,T)$	$\mu_{gp}(fl,P)$, $\mu_{gt}(fl,T)$
Dynamic Viscosity	$\nu_{tp}(fl,T,P)$	$\nu_{fp}(fl,P)$, $\nu_{ft}(fl,T)$	$\nu_{gp}(fl,P)$, $\nu_{gt}(fl,T)$
Thermal Conductivity	$k_{tp}(fl,T,P)$	$k_{fp}(fl,P)$, $k_{ft}(fl,T)$	$k_{gp}(fl,P)$, $k_{gt}(fl,T)$

** The parameter, fl, is the fluid string required by the RefProp functions.*

Co-existing Phase Functions

Property	Function	Property	Function
Saturation Temperature	$T_{sat}(fl,P)$	Vapor Quality	$X_{ph}(fl,P,h)$, $X_{ps}(fl,P,s)$, $X_{pp}(fl,P,\rho)$, $X_{pu}(fl,P,u)$, $X_{th}(fl,T,h)$, $X_{ts}(fl,T,s)$, $X_{tp}(fl,T,\rho)$, $X_{tu}(fl,T,u)$
Saturation Pressure	$P_{sat}(fl,T)$		
Surface Tension	$\sigma_t(fl,T)$		
Density Difference	$\Delta\rho_t(fl,T)$ $[=\rho_l-\rho_g]$		
Enthapy of Vaporization	$h_{fg}(fl,T)$ $[=h_f-h_g]$		

Additional Properties

Isentropic Exponent, γ	$\gamma(fl,T,P)$
Coeff. Thermal Expan.	$\beta(fl,T,P)$
Compressibility	$Z_{tp}(fl,T,P)$, $Z_c(fl,0)$, $Z_{ft}(fl,T)$, $Z_{fp}(fl,P)$, $Z_{gt}(fl,T)$, $Z_{gp}(fl,P)$,
Morton Number	Mo(fl,P)
Kinematic Viscosity	$\nu_{tp}(fl,T,P)$, $\nu_{ft}(fl,T)$,…etc.
Prandtl Number	$Pr_{tp}(fl,T,P)$, $Pr_{ft}(fl,T)$,…etc.

Reverse Functions

$T_{ph}(fl,p,h)$	$T_{hs}(fl,h,s)$
$T_{ps}(fl,p,s)$	$P_{hs}(fl,h,s)$
$P_{th}(fl,t,h)$	$P_{ts}(fl,t,s)$
$\rho_{th}(fl,t,s)$	$\rho_{ts}(fl,t,s)$
$\rho_{ph}(fl,p,h)$	$\rho_{ps}(fl,p,s)$
$h_{ps}(fl,p,s)$	$s_{ph}(fl,p,h)$
$h_{ts}(fl,t,s)$	$s_{th}(fl,t,h)$

Property Constants

Critical Point	$T_c(fl,0)$, $P_c(fl,0)$, $\rho_c(fl,0)$
Triple Point	$T_t(fl,0)$, $P_t(fl,0)$, $\rho_t(fl,0)$
Molecular Weight	$MW(fl,0)$
Ideal Gas Constant	$R_g(fl,0)$

NOTE: The dummy argument, 0, in each of these functions is a placeholder for the eventual implementation of mixtures.

Chapter 5 Reverse Functions

Since all of the ***RefProp*** functions rely on the Helmholtz free energy surface (***f***), they are all constructed as direct functions of Temperature and Density (just like ***f***). While every function of Temperature and Pressure starts with a reverse function to calculate density (so that the direct functions may be used), there are some additional functions that are of use in standard thermodynamic energy cycle calculations. These functions require more complex iteration and convergence behavior and so may require more computational time to evaluate. This should be kept in mind when using these functions, especially if called repeatedly within a loop.

Additionally, some of the functions below are multi-valued; specifically the Pressure, Density, and Entropy functions of Temperature and Enthalpy. This occurs because the curves may fold back on themselves (***Figure 1***); creating an upper and lower density intersection point for a given input (H) parameter (see Figure 1 below). In these situations, there is an extra parameter, ***r***, in the parameter list that indicates whether the lower density root (***r=1***), or the upper root (***r=2***) is requested.

The reverse functions are categorized below by return value. The first parameter in all of these functions is a "**fluid string**" variable that identifies the fluid material to be used for the property calculations.

Following the standard notation, the reverse function names all begin with ***rp_*** followed by the returned property, which is followed by the two input parameters in the order expected in the parameter list.

A list of the various reverse functions is shown below.

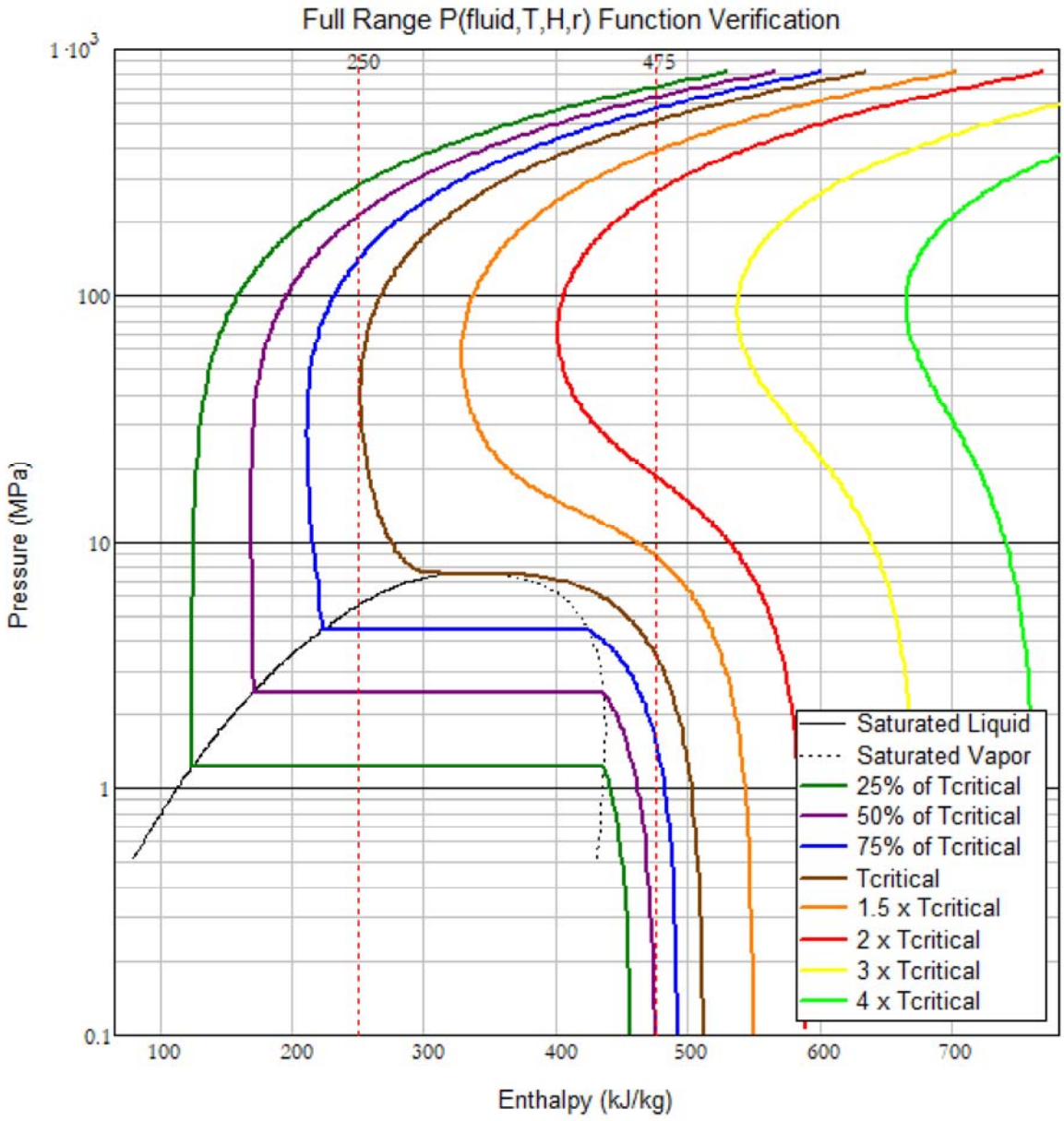


Figure 1 : *Pressure-Enthalpy plot showing double valued isotherms for given values of Enthalpy, H. The higher density roots usually occur at very high pressures, way above the critical point. However, at low enthalpy and temperature values, dual roots can occur in the sub-cooled liquid region.*

Temperature Functions, t

Raw Function	Application of Units
$\text{rp\_tph}(fl,t,h)$	$T_{ph}(fl,p,h) := \text{rp\_tph}\left(fl, \frac{p}{MPa}, h \cdot \frac{kg}{kJ}\right) \cdot K$
$\text{rp\_tps}(fl,p,s)$	$T_{ps}(fl,p,s) := \text{rp\_tps}\left(fl, \frac{p}{MPa}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot K$
$\text{rp\_ths}(fl,h,s)$	$T_{hs}(fl,h,s) := \text{rp\_ths}\left(fl, h \cdot \frac{kg}{kJ}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot K$

Pressure Functions, p

Raw Function	Application of Units
$\text{rp\_pth}(fl,t,h,r)$	$P_{th}(fl,t,h,r) := \text{rp\_pth}\left(fl, \frac{t}{K}, h \cdot \frac{kg}{kJ}, r\right) \cdot MPa$
$\text{rp\_pts}(fl,t,s)$	$P_{ts}(fl,t,s) := \text{rp\_pts}\left(fl, \frac{t}{K}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot MPa$
$\text{rp\_phs}(fl,h \cdot s)$	$P_{hs}(fl,h,s) := \text{rp\_phs}\left(fl, h \cdot \frac{kg}{kJ}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot MPa$

Density Functions, rho

Raw Function	Application of Units
$\text{rp\_rhoth}(fl,t,h \cdot r)$	$\rho_{th}(fl,t,h,r) := \text{rp\_rhoth}\left(fl, \frac{t}{K}, h \cdot \frac{kg}{kJ}, r\right) \cdot \frac{kg}{m^3}$
$\text{rp\_rhoph}(fl,p,h)$	$\rho_{ph}(fl,p,h) := \text{rp\_rhoph}\left(fl, \frac{p}{MPa}, h \cdot \frac{kg}{kJ}\right) \cdot \frac{kg}{m^3}$
$\text{rp\_rhots}(fl,t,s)$	$\rho_{ts}(fl,t,s) := \text{rp\_rhots}(fl,t,s) \cdot \frac{kg}{m^3}$
$\text{rp\_rhops}(fl,p,s)$	$\rho_{ps}(fl,p,s) := \text{rp\_rhops}\left(fl, \frac{p}{MPa}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kg}{m^3}$

Enthalpy Functions, h

Raw Function	Application of Units
$\text{rp\_hps}(fl, p, s)$	$h_{ps}(fl, p, s) := \text{rp\_hps}\left(fl, \frac{p}{MPa}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kJ}{kg}$
$\text{rp\_hts}(fl, t, h)$	$h_{ts}(fl, t, h) := \text{rp\_hts}\left(fl, \frac{t}{K}, h \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kJ}{kg}$

Entropy Functions, s

Raw Function	Application of Units
$\text{rp\_sph}(fl, p, h)$	$s_{ph}(fl, p, h) := \text{rp\_sph}\left(fl, \frac{p}{MPa}, h \cdot \frac{kg}{kJ}\right) \cdot \frac{kJ}{kg \cdot K}$
$\text{rp\_sth}(fl, t, h, r)$	$s_{th}(fl, t, h, r) := \text{rp\_sth}\left(fl, \frac{t}{K}, h \cdot \frac{kg}{kJ}, r\right) \cdot \frac{kJ}{kg \cdot K}$

Chapter 6 Utility Functions

Various utility functions are also provided through the Mathcad ***RefProp*** interface. These routines are provided for verification of the loaded material files, retrieval of material parameters, and retrieval of the version of the NIST ***RefProp*** library that has been loaded.

Mathcad RefProp Interface Add-in Version

This function returns a string containing the version number of the Mathcad ***RefProp*** interface DLL. This routine is provided for version control, and to ensure that specific calculations are prepared and viewed with the same version of the Add-In DLL.

Function Definition

$\text{rp\_getvers}(dummy)$ *NOTE: The dummy parameter can by any integer, i.e. 0.*

Example

$\text{rp\_getvers}(0) = \text{“RefProp Mathcad Add-in: Version 2.0”}$

NIST RefProp Library Version

This function returns a string containing the version number of the ***NIST RefProp*** library. This string is extracted from the database file location auto-selected by the Mathcad ***RefProp*** Add-in or the environment variable, ***NIST_PATH***.

Function Definition

$\text{rp\_getNIST}(dummy)$ *NOTE: The dummy parameter can by any integer, i.e. 0.*

Example

$\text{rp\_getNIST}(0) = \text{“NIST RefProp Library: Version 10.0.0.0”}$

Path to Loaded REFPROP.DLL

For debugging purposes, it can be helpful to know where Mathcad found the REFPROP dynamic linked library (DLL). This function shows the full path to the DLL that was loaded when Mathcad initialized the RefProp Add-in functions.

$$\text{rp\_getpath}(0) = \text{“C:\Program Files (x86)\REFPROP\REFPRP64.dll”}$$

Note that if Mathcad cannot find the REFPROP (DLL), a the user is warned with a pop-up message and the RefProp Add-in functions will not be loaded and registered into the Mathcad interface.

Ideal Gas Constant, $\mathbf{R_{gas}}$

For some fluid materials, the value of the ideal gas constant may vary slightly. This utility function will report the Ideal Gas Constant, R_{gas}, in units of ***J/mol-K*** as listed for a specific "**fluid string**".

Function Definition

$$\text{rp\_rgas}(fluid, comp)$$

Application of Units

$$R_{gas}(fluid, comp) := \text{rp\_rgas}(fluid, comp) \cdot \frac{J}{mol \cdot K}$$

NOTE: The additional parameter, ***comp****, is a component number for a mixture. Since mixtures are not yet implemented, this number should be 1 or 0.*

Example

$$R_{gas}(\text{“WATER.fld”}, 0) = 8.3144 \ \frac{J}{mol \cdot K}$$

Molecular Weight,

This function extracts the molecular weight of the material specified by the "**fluid string**", in units of ***gm/mol***.

Function Definition

$\text{rp\_wmol}(fl, comp)$

Application of Units

$\text{WM}(fl, comp) := \text{rp\_wmol}(fl, comp) \cdot \frac{gm}{mol}$

*NOTE: The additional parameter, **comp**, is a component number for a mixture. Since mixtures are not yet implemented, this number should be 1 or 0.*

Example

$\text{WM}(\text{“WATER.FLD”}, 0) = 18.0153 \ \frac{gm}{mol}$

$\text{WM}(\text{“CO2.fld”}, 0) = 44.0098 \ \frac{gm}{mol}$

Material Full Name

This function returns a string containing the full name of the material specified by the "**fluid string**". This can provide useful verification that the correct fluid string is being used.

Function Definition

$\text{rp\_getname}(fl, comp)$

*NOTE: The additional parameter, **comp**, is a component number for a mixture. Since mixtures are not yet implemented, this number should be 1 or 0.*

Example

$\text{rp\_getname}(\text{“D2.fld”}, 1) = \text{“Deuterium”}$

$\text{rp\_getname}(\text{“R12.fld”}, 1) = \text{“Dichlorodifluoromethane”}$

Material Chemical Abstracts Serial Number, CASN

This function returns a string containing the Chemical Abstracts Serial Number, ***CASN***, of the material specified by the "**fluid string**"**.** This can provide useful verification that the correct fluid string is being used.

Function Definition

$\text{rp\_getcasn}(fl, comp)$

NOTE: The additional parameter, ***comp****, is a component number for a mixture. Since mixtures are not yet implemented, this number should be 1 or 0.*

Example

$\text{rp\_getcasn}(\text{“D2.fld”}, 1) = \text{“7782-39-0”}$

$\text{rp\_getcasn}(\text{“R12.fld”}, 1) = \text{“75-71-8”}$

Extrapolation Flag

The ***NIST RefProp*** database allows all functions to be extrapolated out to 150% of the maximum temperature and 200% of the maximum pressure values specified in the material files. This behavior has been found to be fairly accurate, since the curves are well behaved, if not linear, beyond the experimental temperature/ pressure limits in the super-critical region.

The rp_extrap function sets a flag that allows extrapolation out to *1.5 x* T_{max} and *2.0 x* P_{max} , and returns a verification string as to the state of the flag. Once set, this flag is used for all calculations below and to the right of the call, or until another call is made. ***Extrapolation is OFF by default***.

Function Definition

$\text{rp\_extrap}(flag)$

flag: 0 = no extrapolation past T_{max} *(default)*
1 = extrapolate to $1.5 \cdot T_{max}$.
extrapolate to $2.0 \cdot P_{max}$

Example

$\text{rp\_extrap}(0) = \text{“Extrapolation Disabled”}$

$\text{rp\_extrap}(1) = \text{“Extrapolation Enabled”}$

Chapter 7 Mixture Properties

In addition to selecting pure fluids, RefProp has very sophisticated algorithms for calculating properties of mixtures. There are several ways to specify a mixture as the requested fluid in RefProp:

- **Pseudo-Pure Fluids**
- **Predefined Mixture Files**
- **Custom Mixture Files**
- **Ad-Hoc Fluid Mixture Strings** (*provided by the Mathcad add-in*)

These methods will be discussed in detail below.

Pseudo-Pure Fluids

There are a few mixtures for which the mixture properties have been curve fit as though the mixture was a pure fluid. These are pseudo-pure fluids

Details:

- The "**fluid string**" can be set to one of the pseudo-pure fluids files to be loaded from the "fluids\" directory. These files MUST end with the extension "**.ppf**".

e.g. to load the properties for AIR, define a variable such as:

fluid := "AIR.ppf"

- A list of the pseudo-pure fluid files available as of REFPROP 9.1 are are shown in the table below[1].

Pseudo-Pure Fluids use extension ".ppf"				
Air	R404A	R407A	R410A	R507A

[1] Current REFPROP version documentation should be consulted for the complete list

Predefined Mixture Files

Mixture files are small files that identify the component pure fluids that make up the mixture and the single-phase mole fractions of each component. The file also contains a number of parameters to provide the mixture molecular mass and the limiting temperatures and pressures for the overall mixture.

For example, this is the mixture file for Air (as opposed to the .ppf file described above). This text file, AIR.mix, is in the [REFPROP]\Mixtures directory.

```
Air (dry)
 28.958600656 132.791424977327 3844.70114021364 11.9051814519786
 3
NITROGEN.FLD
ARGON.FLD
OXYGEN.FLD
 .7812
 .0092
 .2096
0
```

The table below contains a list of all of the predefined mixture files available as of RefProp 9.1, stored in the mixtures folder of the REFPROP installation directory. The vast majority of these mixture files are refrigerant combinations in addition to AIR and a handful of regional natural gas mixtures[2].

Pre-defined Mixtures use extension ".mix"				
R401A	R408A	R419A	R431A	R503
R401B	R409A	R420A	R432A	R504
R401C	R409B	R421A	R433A	R507A
R402A	R410A	R421B	R434A	R508A
R402B	R410B	R422A	R435A	R508B
R403A	R411A	R422B	R436A	R509A
R403B	R411B	R422C	R436B	R510A
R404A	R412A	R422D	R437A	R512A
R405A	R413A	R423A	R438A	AIR
R406A	R414A	R424A	R441A	AMARILLO
R407A	R414B	R425A	R442A	EKOFISK
R407B	R415A	R426A	R443A	GLFCOAST
R407C	R415B	R427A	R444A	HIGHCO2
R407D	R416A	R428A	R500	HIGHN2
R407E	R417A	R429A	R501	
R407F	R418A	R430A	R502	

[2] Current REFPROP version documentation should be consulted for the complete list

Details:

- The "**fluid string**" can be set to one of the pre-defined mixture files to be loaded from the "mixtures\" directory. These files MUST end with the extension "**.mix**".

e.g. to load the properties for AIR, define a variable such as:

$fluid := \text{“AIR.mix”}$

Custom Mixture Files

Since mixture files are just text files, new mixtures of pure fluids can be created. REFPROP will apply its mixture correlations to provide fairly accurate mixture properties.

The format of a mixture file is as follows,

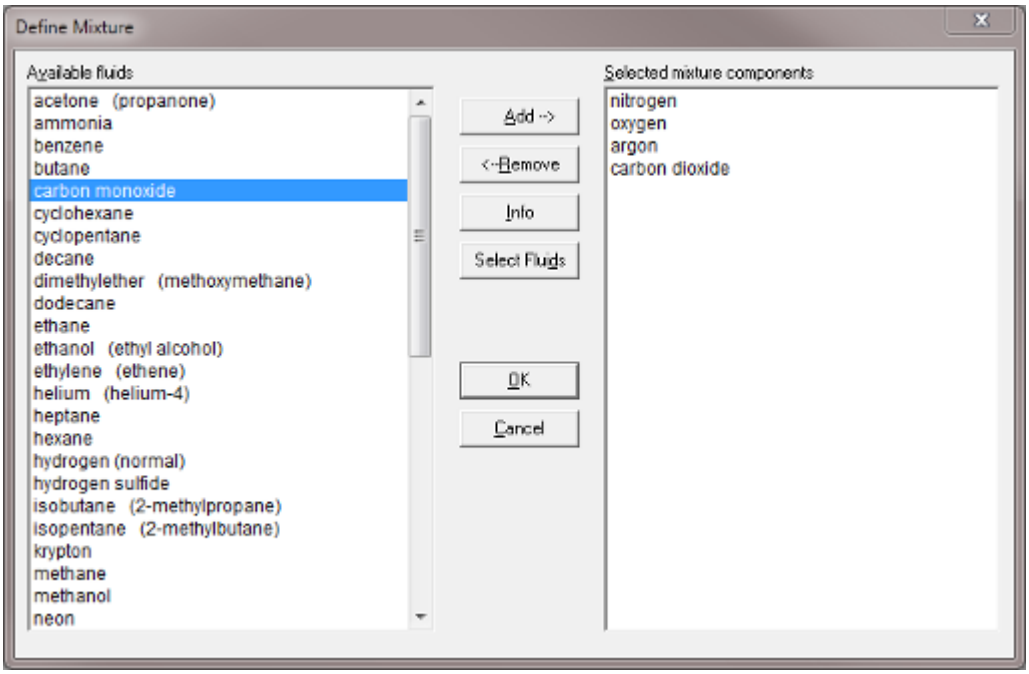
```
Mixture Name
 [molar mass] [Tcrit(K)] [Pcrit(kPa)] [Dcrit(mol/m³)]
 n                // = number of components
fluid1.fld        // component 1
fluid2.fld        // component 2
 |
fluidn.fld        // component n
 mf1              // mole fraction 1
 mf2              // mole fraction 2
 |
 mfn              // mole fraction n
0                 // end of file delimiter
```

These files can be constructed for new mixtures so that they can be reused over and over again, without having to specify the mixture parameters repeatedly. Luckily, the Graphical User Interface (GUI) for RefProp provides a handy facility for creating and saving new mixture files[3].

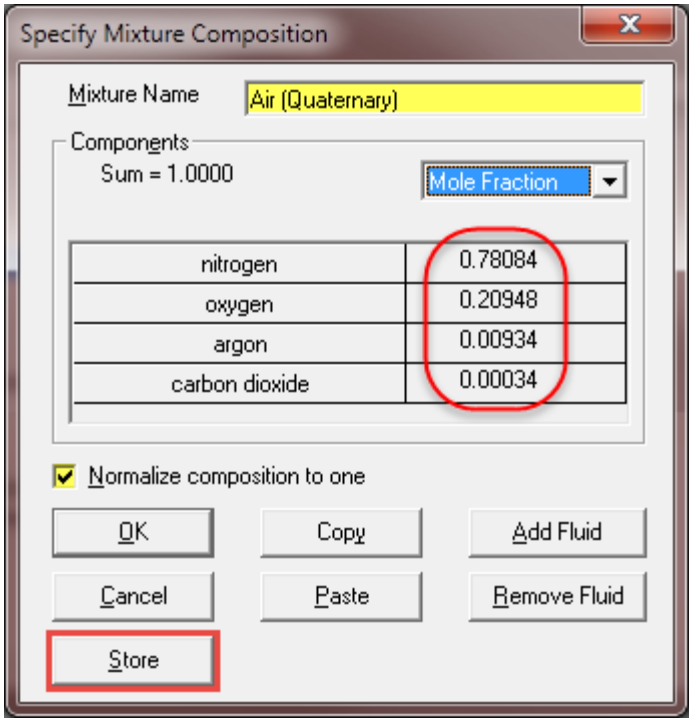
As an example, let's create an alternative mixture for air that has slightly different mole fractions and an additional carbon dioxide component than the predefined AIR.mix file.

[3] The [REFPROP]\mixtures folder should not be read-only and the user must have write access permission for this method to work.

Open the REFPROP GUI and select New Mixture from the Substance menu. In the Define Mixture panel, select the four components for air, including carbon dioxide, to form a quaternary mixture.



Select OK and then fill in the mole fractions for each component per the figure below. All component mole fractions should sum to 1.0; however, you can select the checkbox to normalize the mole fractions to one and REFPROP will do just that. Change the name in the top box to a suitable name for the mixture.



Select Store and store the mixture file in the mixtures directory with an appropriate file name (like AIR4.MIX). This mixture file will now be available for use in REFPROP.

Let's use the original and new mixture files for air to calculate room temperature density and compare it to the pseudo-pure fluid value.

$T := 22\ \mathbf{°C}$ $P := 1 \cdot \mathbf{atm}$

$$\rho_{orig} := \text{rp\_rhotp}\left(\text{“AIR.MIX”}, \frac{T}{\mathbf{K}}, \frac{P}{\mathbf{MPa}}\right) \cdot \frac{\mathbf{kg}}{\mathbf{m}^3} = 1.196079\ \frac{kg}{m^3}$$

$$\rho_{AIR4.MIX} := \text{rp\_rhotp}\left(\text{“AIR4.mix”}, \frac{T}{\mathbf{K}}, \frac{P}{\mathbf{MPa}}\right) \cdot \frac{\mathbf{kg}}{\mathbf{m}^3} = 1.196354\ \frac{kg}{m^3}$$

$$\rho_{ppf} := \text{rp\_rhotp}\left(\text{“AIR.ppf”}, \frac{T}{\mathbf{K}}, \frac{P}{\mathbf{MPa}}\right) \cdot \frac{\mathbf{kg}}{\mathbf{m}^3} = 1.196397\ \frac{kg}{m^3}$$

$$\frac{\rho_{orig} - \rho_{ppf}}{\rho_{ppf}} = -0.0265\ \mathbf{\%} \qquad \frac{\rho_{AIR4.MIX} - \rho_{ppf}}{\rho_{ppf}} = -0.0036\ \mathbf{\%}$$

It appears that the new quaternary mixture including carbon dioxide is closer to the pseudo-pure fluid density than the ternary mixture. Both mixture densities, however, are fairly close to the density calculated from the pseudo-pure fluid.

Ad-Hoc Mixture Strings

REFPROP provides facility for embedding multiple mixture components right in the fluid string passed to the interface functions. The Mathcad Add-In interface, takes this a step further and allows not only the component names, but also the *mole fractions* of each component to be embedded in the fluid string. The format for mixture fluids is as follows. Within the fluid string,

1. Separate each component with an "**&**" character,
2. Follow each component with its mole fraction, surrounded by "**[**" and "**]**",
3. The number of components is limited to ***20***, although the more components makes the flash calculations harder and they may fail in the REFPROP solver,
4. Zero mole fraction components will be parsed out,
5. Total of mole fractions must sum exactly to one (**1.0**).

For example, a binary representation for Air could be input with the following fluid string.

$$Air_{binary} := \text{“Oxygen[0.21]\&Nitrogen[0.79]”}$$

$$\rho_{air2} := \text{rp\_rhotp}\left(Air_{binary}, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kg}{m^3} = 1.191606 \ \frac{kg}{m^3}$$

$$\rho_{ppf} = 1.196397 \ \frac{kg}{m^3} \qquad \frac{\rho_{ppf} - \rho_{air2}}{\rho_{ppf}} = 0.4 \ \%$$

This binary representation for air carries a bit more error than the more detailed mixtures shown previously. The mixture string, however, can hold up to 20 components, allowing the mixture representation to be as accurate as needed, including many of the trace components of air. Failure to adhere to the mixture string format will result in a Mathcad error on the function call.

To facilitate this process, a utility function, $\text{MixString}(M)$ is provided in the ***Refprop_units*** reference worksheet that simplifies assembling these mixture strings and making sure that the mole fractions all sum to unity.

To use this utility user function, first include the reference worksheet:

Include << .\RefProp_units.mcdx

Set up the representation for the mixture in a matrix, where the first column is the pure component file string and the second column is the corresponding mole fraction of each component. Check that the second column sums to 1.0.

$$M2 := \begin{bmatrix} \text{“Oxygen”} & 0.21 \\ \text{“Nitrogen”} & 0.79 \end{bmatrix} \qquad \sum M2^{\langle 1 \rangle} = 1.000000000000000$$

$$M3 := \begin{bmatrix} \text{“Oxygen”} & 0.2096 \\ \text{“Nitrogen”} & 0.7812 \\ \text{“Argon”} & 0.0092 \end{bmatrix} \qquad \sum M3^{\langle 1 \rangle} = 1.000000000000000$$

$$M4 := \begin{bmatrix} \text{“Oxygen”} & 0.20948 \\ \text{“Nitrogen”} & 0.78084 \\ \text{“Argon”} & 0.00934 \\ \text{“CO2”} & 0.00034 \end{bmatrix} \qquad \sum M4^{\langle 1 \rangle} = 1.000000000000000$$

Now the mixture strings can be assembled using the MixString utility function.

$$Air2 := \text{MixString}(M2) = \text{“Oxygen[0.21]\&Nitrogen[0.79]”}$$

$$Air3 := \text{MixString}(M3) = \text{“Oxygen[0.2096]\&Nitrogen[0.7812]\&Argon[0.0092]”}$$

$$Air4 := \text{MixString}(M4) = \text{“Oxygen[0.20948]\&Nitrogen[0.78084]\&Argon[0.00934]\&CO2[0.00034]”}$$

The property functions, $\rho_{tp}(\text{“fluid”}, T, P)$, from the unit reference worksheet can then be called using this utility function directly.

$$\rho_{air2} := \rho_{tp}(\text{MixString}(M2), T, P) = 1.191606 \ \frac{kg}{m^3} \qquad \frac{\rho_{ppf} - \rho_{air2}}{\rho_{ppf}} = 0.4 \ \%$$

$$\rho_{air3} := \rho_{tp}(\text{MixString}(M3), T, P) = 1.196079 \ \frac{kg}{m^3} \qquad \frac{\rho_{ppf} - \rho_{air3}}{\rho_{ppf}} = 0.027 \ \%$$

$$\rho_{air4} := \rho_{tp}(\text{MixString}(M4), T, P) = 1.196354 \ \frac{kg}{m^3} \qquad \frac{\rho_{ppf} - \rho_{air4}}{\rho_{ppf}} = 0.004 \ \%$$

The tertiary ad-hoc mixture gives exactly the same result as the predefined mixture file (*AIR.MIX*), since the ad-hoc string contains exactly the same components and mole fractions. The quaternary mixture results in a more accurate density calculation, compared to the pseudo-pure fluid (*AIR.PPF*) density, and exactly the same result, $\rho_{AIR4.MIX} = 1.196354 \ kg \cdot m^{-3}$, as was calculated using the custom air mixture file (*AIR4.MIX*) above.

Appendix A Fluid Property Functions

RefProp is based on the most accurate pure fluid and mixture models currently available. It implements three models for the thermodynamic properties of pure fluids: equations of state explicit in Helmholtz energy, the modified Benedict-Webb-Rubin equation of state, and an extended corresponding states (ECS) model.

Mixture calculations employ a model that applies mixing rules to the Helmholtz energy of the mixture components; it uses a departure function to account for the departure from ideal mixing.

Thermodynamic Properties

The underlying NIST routines are provided to calculate thermodynamic and transport properties at a given (T,ρ,x) state. The functions below, are provided as functions of Temperature (T) and Pressure (P). This requires an iterative calculation of Density (ρ) followed by a direct calculation of the requested property in terms of Temperature and Density.

Transport Properties

Viscosity and thermal conductivity are modeled with either fluid-specific correlations, an ECS (extended corresponding states) method, or in some cases the friction theory method. Surface tension is calculated from curve fit correlations specific to each fluid material.

5.9	**Surface Tension, σ**
5.10	**Thermal Conductivity, k**
5.11	**Viscosity, μ**

See also

5.0	**Reverse Functions**
6.0	**Utility Functions**

Some of the examples in this appendix require the units reference from Appendix C.

Include << .\RefProp_units.mcdx

A.1 Density, ρ

The fluid density describes the mass of a unit volume of fluid at a given state and is expressed in units of $kg \cdot m^{-3}$. Since the Helmholtz equation is a function of temperature and density, calculating density in terms of temperature and pressure requires an iterative solution.

solved equation: $$p = \rho^2 \cdot \left(\frac{\mathrm{d}}{\mathrm{d}\rho} f \right)_T$$

Specific volume can be evaluated as the inverse of the density.

This iterative density calculation provides the basis for all other thermodynamic property calculations, which can be evaluated directly in terms of Temperature and Density.

Functions that return Density (rho)

rp_rhotp(*fluid*, *t*, *p*)	Obtains density of ***superheated vapor*** or ***compressed liquid*** in kg/m³ as a function of temperature (t) in K and Pressure (p) in MPa.
rp_rhoft(*fluid*, *t*)	Obtains density of ***saturated liquid*** in kg/m³ as a function of temperature (t) in K.
rp_rhofp(*fluid*, *p*)	Obtains density of ***saturated liquid*** in kg/m³ as a function of pressure (p) in MPa.
rp_rhogt(*fluid*, *t*)	Obtains density of ***saturated vapor*** in kg/m³ as a function of temperature (t) in K.
rp_rhogp(*fluid*, *p*)	Obtains density of ***saturated vapor*** in m³/kg as a function of pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a "**fluid string**" variable that identifies the fluid material to be used for the property calculations.

Application of Units

Mathcad user functions are created here to calculate specific volume ($1/\rho$) with the application of units.

$$v_{tp}(fluid,t,p) := \left(\text{rp\_rhotp}\left(fluid,\frac{t}{K},\frac{p}{MPa}\right)\cdot\frac{kg}{m^3}\right)^{-1}$$

$$v_{ft}(fluid,t) := \left(\text{rp\_rhoft}\left(fluid,\frac{t}{K}\right)\cdot\frac{kg}{m^3}\right)^{-1} \qquad v_{gt}(fluid,t) := \left(\text{rp\_rhogt}\left(fluid,\frac{t}{K}\right)\cdot\frac{kg}{m^3}\right)^{-1}$$

$$v_{fp}(fluid,p) := \left(\text{rp\_rhofp}\left(fluid,\frac{p}{MPa}\right)\cdot\frac{kg}{m^3}\right)^{-1} \qquad v_{gp}(fluid,p) := \left(\text{rp\_rhogp}\left(fluid,\frac{p}{MPa}\right)\cdot\frac{kg}{m^3}\right)^{-1}$$

Example

$fl := \text{"WATER.fld"}$

$T_c := \text{rp\_tcrit}(fl,0)\ K = 705.103\ °F$

$T_t := \text{rp\_ttrip}(fl,0)\ K = 32.018\ °F$

Temperatures

$n := 200$

$\Delta T := (T_c - T_t)\cdot\frac{1}{n}$

$t1 := T_t, T_t + \Delta T .. 2\cdot T_c$

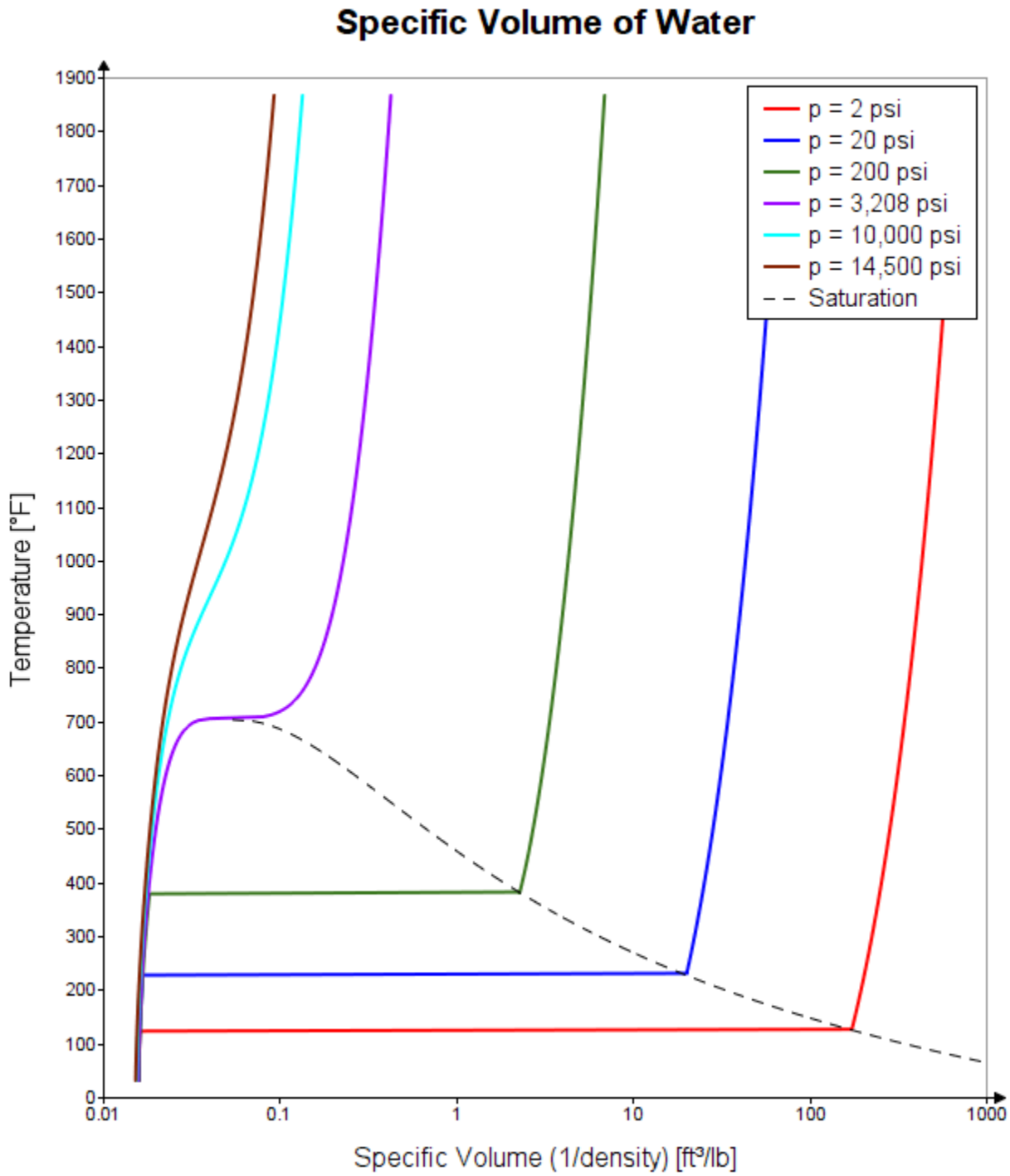
Pressures

$p1 := 2\cdot psi$ $\quad p4 := 3208\cdot psi$

$p2 := 20\cdot psi$ $\quad p5 := 10000\cdot psi$

$p3 := 200\cdot psi$ $\quad p6 := 14500\cdot psi$

The plot of specific volume vs. temperature at the specified pressures on the following page provides a continuity check for these functions.


Specific Volume of Water
p = 2 psi
p = 20 psi
p = 200 psi
p = 3,208 psi
p = 10,000 psi
p = 14,500 psi
Saturation
Temperature [°F]
1900
1800
1700
1600
1500
1400
1300
1200
1100
1000
900
800
700
600
500
400
300
200
100
0
0.01
0.1
1
10
100
1000
Specific Volume (1/density) [ft³/lb]

A.2 Specific Internal Energy, u

Internal energy represents the total energy stored in a fluid. It is the sum of the molecular kinetic energy and the molecular kinetic energy within the fluid and has the units of kJ/kg. The internal energy of a substance does not include any energy that it may posses as a result of its position or movement as a whole. Rather it refers to the energy of the molecules making up the substance.

Internal energy is calculated from the Helmholtz free energy surface, $f(\rho, T)$, as follows..

Internal Energy: $$u = f - T \cdot \left(\frac{\mathrm{d}}{\mathrm{d}T} f \right)_{\rho}$$

The total internal energy of a fluid can not be measured. However, changes in internal energy calculated from a reference state can be evaluated. For water, this reference state is saturated liquid at 0.01°C (273.16 K). At the reference state, internal energy, u, is set to zero (as demonstrated in the example below).

Evaluate at reference state: $T_{ref} := 273.16 \cdot K$ $fluid := \text{“WATER.fld”}$

$$\text{rp\_uft}\left(fluid, \frac{T_{ref}}{K}\right) \cdot \frac{kJ}{kg} = 0.00000000 \ \frac{kJ}{kg}$$

Functions that return Internal Energy

$rp\_utp(fluid, t, p)$	Obtains internal energy of ***superheated vapor*** or ***compressed liquid*** in kJ/kg as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_uft(fluid, t)$	Obtains internal energy of ***saturated liquid*** in kJ/kg as a function of temperature (t) in K.
$rp\_ufp(fluid, p)$	Obtains internal energy of ***saturated liquid*** in kJ/kg as a function of pressure (p) in MPa.

$rp\_ugt(fluid, t)$	Obtains internal energy of ***saturated vapor*** in kJ/kg as a function of temperature (t) in K.
$rp\_ufp(fluid, p)$	Obtains internal energy of ***saturated liquid*** in kJ/kg as a function of pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a "**fluid string**" variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$u_{tp}(fluid, t, p) := \text{rp\_utp}\left(fluid, \frac{t}{K}, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg}$$

$$u_{ft}(fluid, t) := \text{rp\_uft}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg} \qquad u_{gt}(fluid, t) := \text{rp\_ugt}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg}$$

$$u_{fp}(fluid, p) := \text{rp\_ufp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg} \qquad u_{gp}(fluid, p) := \text{rp\_ugp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg}$$

Example

$fl := \text{“WATER.fld”}$

Temperatures

$n := 200$

$\Delta T := (T_c - T_t) \cdot \frac{1}{n}$

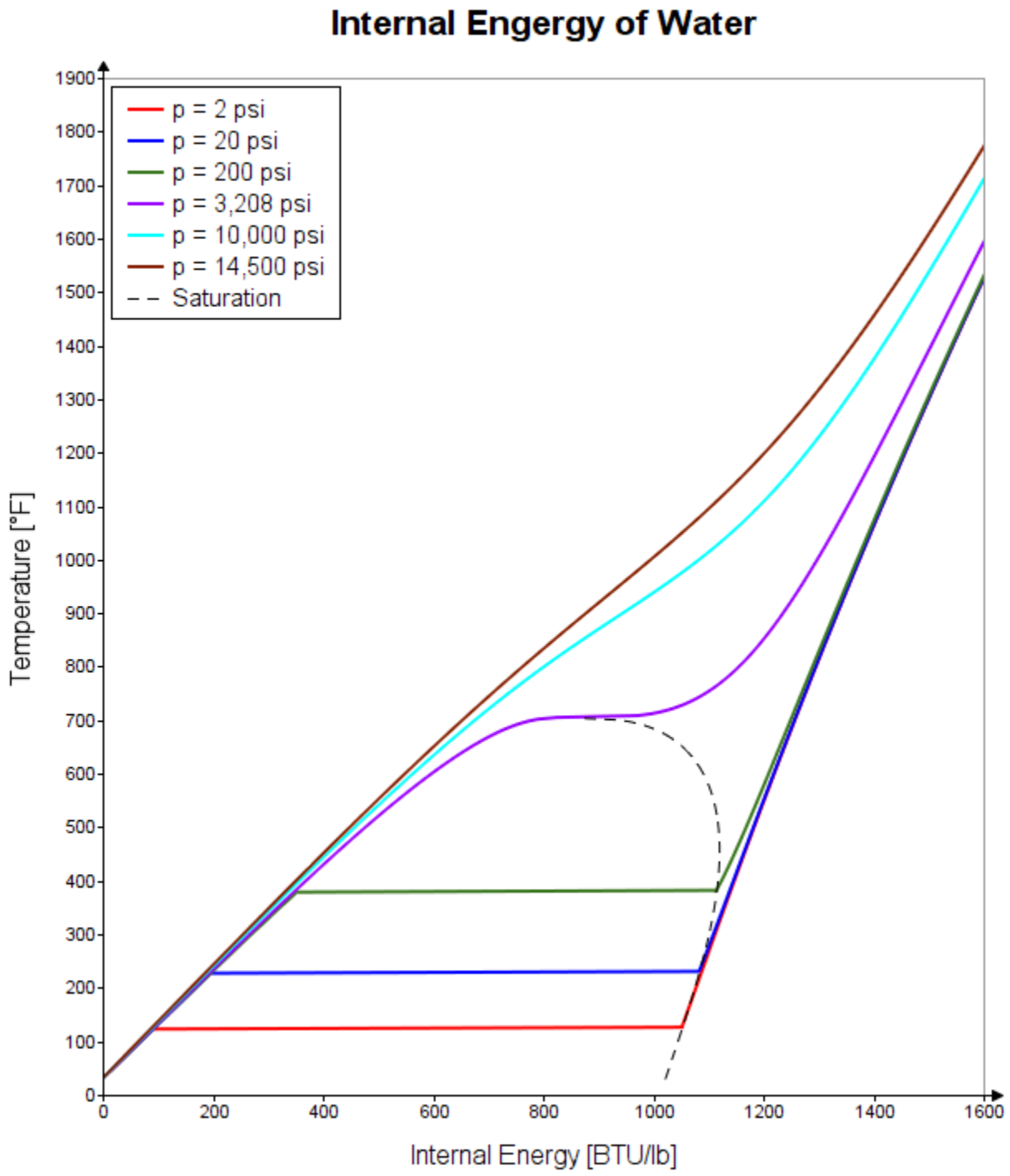
$t1 := T_t, T_t + \Delta T .. 2 \cdot T_c$

Pressures

$p1 := 2 \cdot psi$ $\quad p4 := 3208 \cdot psi$

$p2 := 20 \cdot psi$ $\quad p5 := 10000 \cdot psi$

$p3 := 200 \cdot psi$ $\quad p6 := 14500 \cdot psi$


Internal Engergy of Water
p = 2 psi
p = 20 psi
p = 200 psi
p = 3,208 psi
p = 10,000 psi
p = 14,500 psi
Saturation
Temperature [°F]
Internal Energy [BTU/lb]
0
100
200
300
400
500
600
700
800
900
1000
1100
1200
1300
1400
1500
1600
1700
1800
1900
0
200
400
600
800
1000
1200
1400
1600

A.3 Specific Enthalpy, h

Enthalpy is defined (out of convenience) as the sum of the internal energy, u and the pressure-volume product, $P \cdot v$ and has the units of kJ/kg:

$$h = u + P \cdot v$$

Internally, enthalpy is calculated from the Helmholtz free energy surface, $f(\rho, T)$, in the following manner:

specific enthalpy: $$h = f - T \cdot \left(\frac{\mathrm{d}}{\mathrm{d}T} f \right)_{\rho} + \rho \cdot \left(\frac{\mathrm{d}}{\mathrm{d}\rho} f \right)_{T}$$

Keep in mind that enthalpy is a property, and its use in calculating internal energy at the same state is not dependent on any processes that may be occurring. Values of enthalpy are relative to an arbitrarily chosen reference state. For water, this reference state is saturated liquid at 0.01°C (273.16 K). At the reference state, internal energy, u, is set to zero, leaving $h = P \cdot v$
(as demonstrated in the example below).

$$fluid := \text{“WATER.fld”}$$

Evaluate at reference state: $$T_{ref} := 273.16 \cdot \boldsymbol{K}$$

$$P_{sat}(fluid, T) := \text{rp\_psatt}\left(fluid, T \cdot \boldsymbol{K}^{-1}\right) \cdot \boldsymbol{MPa}$$

$$h_f(fluid, t) := \text{rp\_hft}\left(fluid, \frac{t}{\boldsymbol{K}}\right) \cdot \frac{kJ}{\boldsymbol{kg}} \qquad v_f(fluid, t) := \left(\text{rp\_rhoft}\left(fluid, \frac{t}{\boldsymbol{K}}\right) \cdot \frac{\boldsymbol{kg}}{\boldsymbol{m}^3}\right)^{-1}$$

$$h_f(fluid, T_{ref}) - P_{sat}(fluid, T_{ref}) \cdot v_f(fluid, T_{ref}) = 0.00000000 \ \frac{kJ}{kg}$$

Functions that return Enthalpy

$rp\_htp(fluid, t, p)$	Obtains enthalpy of ***superheated vapor*** or ***compressed liquid*** in kJ/kg as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_hft(fluid, t)$	Obtains enthalpy of ***saturated liquid*** in kJ/kg as a function of temperature (t) in K.
$rp\_hfp(fluid, p)$	Obtains enthalpy of ***saturated liquid*** in kJ/kg as a function of pressure (p) in MPa.
$rp\_hgt(fluid, t)$	Obtains enthalpy of ***saturated vapor*** in kJ/kg as a function of temperature (t) in K.
$rp\_hgp(fluid, p)$	Obtains enthalpy of ***saturated vapor*** in kJ/kg as a function of pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$h_{tp}(fluid, t, p) := \text{rp\_htp}\left(fluid, \frac{t}{K}, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg}$$

$$h_{ft}(fluid, t) := \text{rp\_hft}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg} \qquad h_{gt}(fluid, t) := \text{rp\_hgt}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg}$$

$$h_{fp}(fluid, p) := \text{rp\_hfp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg} \qquad h_{gp}(fluid, p) := \text{rp\_hgp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg}$$

Example

$fluid$ = "WATER.fld"

Temperatures

$n := 200$

$\Delta T := (T_c - T_t) \cdot \frac{1}{n}$

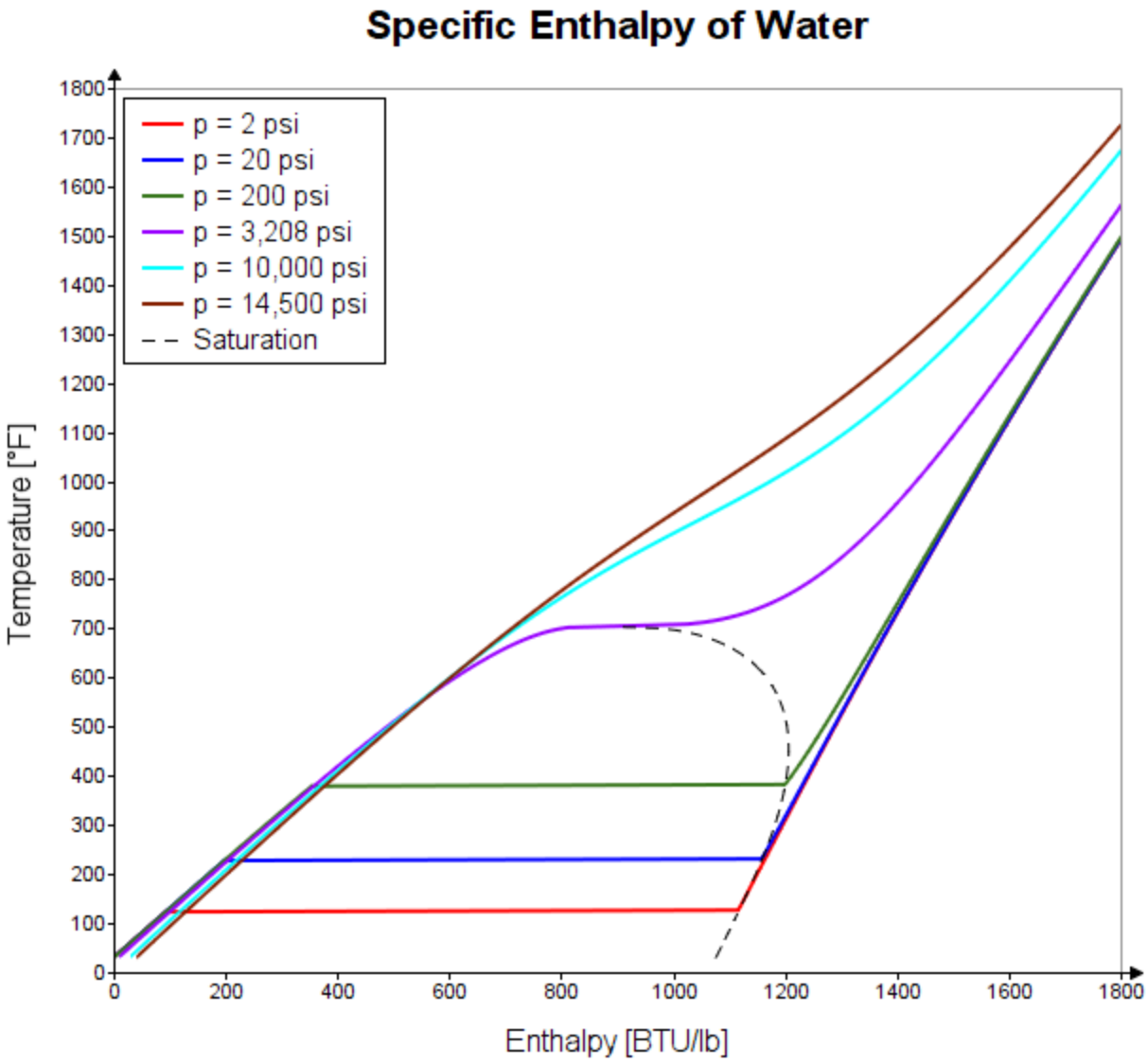
$t1 := T_t, T_t + \Delta T .. 2 \cdot T_c$

Pressures

$p1 := 2 \cdot psi$ $p4 := 3200 \cdot psi$

$p2 := 20 \cdot psi$ $p5 := 10000 \cdot psi$

$p3 := 200 \cdot psi$ $p6 := 14500 \cdot psi$



A.4 Specific Entropy, *s*

For a reversible cyclical process where temperature varies during heat absorption and rejection,

$$\int \frac{1}{T} \mathrm{d}Q = 0 \qquad \text{is true.}$$

Since this is true, it follows that the above integral is path independent. This integral has been defined as the entropy change, s, and has units of kJ/kg-K. The entropy of water is only dependent on its state. Like internal energy, entropy, ***s***, is defined as zero for saturated liquid at 0.01 °C (273.16 K).

The specific entropy of a fluid is determined from the Helmholtz free energy surface, $f(\rho, T)$, by the following equation.

Specific Entropy:

$$s = -\left(\frac{\mathrm{d}}{\mathrm{d}T} f\right)_{\rho}$$

Functions that return Specific Entropy

$rp\_stp(fluid, t, p)$	Obtains entropy of ***superheated vapor*** or ***compressed liquid*** in kJ/kg-K as a function of temperature (t) in K and Pressure (p) in MPa. ***fluid*** is the fluid string for the requested material.
$rp\_sft(fluid, t)$	Obtains entropy of ***saturated liquid*** in kJ/kg-K as a function of temperature (t) in K.
$rp\_sfp(fluid, p)$	Obtains entropy of ***saturated liquid*** in kJ/kg-K as a function of pressure (p) in MPa.
$rp\_sgt(fluid, t)$	Obtains entropy of ***saturated vapor*** in kJ/kg-K as a function of temperature (t) in K.
$rp\_sgt(fluid, p)$	Obtains entropy of ***saturated vapor*** in kJ/kg-K as a function of pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$s_{tp}(fluid,t,p) := \text{rp\_stp}\left(fluid,\frac{t}{K},\frac{p}{MPa}\right)\cdot\frac{kJ}{kg\cdot K} \qquad kJ \equiv 1000\cdot J$$

$$s_{fp}(fluid,p) := \text{rp\_sfp}\left(fluid,\frac{p}{MPa}\right)\cdot\frac{kJ}{kg\cdot K} \qquad s_{ft}(fluid,t) := \text{rp\_sfp}\left(fluid,\frac{t}{K}\right)\cdot\frac{kJ}{kg\cdot K}$$

$$s_{gp}(fluid,p) := \text{rp\_sgp}\left(fluid,\frac{p}{MPa}\right)\cdot\frac{kJ}{kg\cdot K} \qquad s_{gt}(fluid,t) := \text{rp\_sgt}\left(fluid,\frac{t}{K}\right)\cdot\frac{kJ}{kg\cdot K}$$

Example

$fluid :=$ "WATER.fld"

Temperatures

$n := 200$

$\Delta T := (T_c - T_t)\cdot\frac{1}{n}$

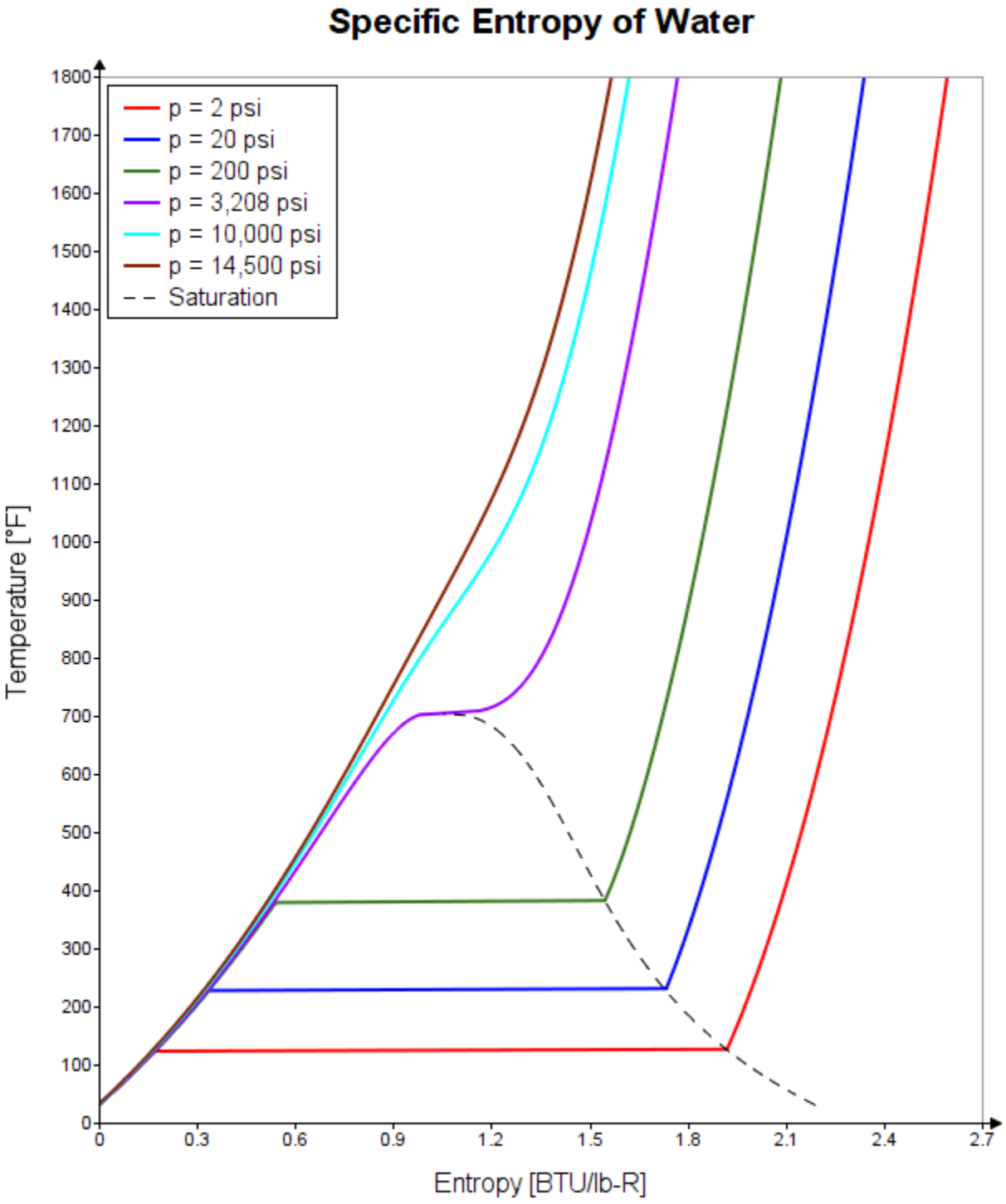
$t1 := T_t, T_t + \Delta T .. 2\cdot T_c$

Pressures

$p1 := 2\cdot psi$ $\quad p4 := 3200\cdot psi$

$p2 := 20\cdot psi$ $\quad p5 := 10000\cdot psi$

$p3 := 200\cdot psi$ $\quad p6 := 14500\cdot psi$


Specific Entropy of Water
p = 2 psi
p = 20 psi
p = 200 psi
p = 3,208 psi
p = 10,000 psi
p = 14,500 psi
Saturation
Temperature [°F]
Entropy [BTU/lb-R]
1800
1700
1600
1500
1400
1300
1200
1100
1000
900
800
700
600
500
400
300
200
100
0
0
0.3
0.6
0.9
1.2
1.5
1.8
2.1
2.4
2.7

A.5 Isobaric Specific Heat, Cp

Isobaric Specific Heat is defined as the rate of change of enthalpy of a unit mass of material with respect to temperature at a given constant pressure (isobaric):

$$c_p = \left(\frac{\mathrm{d}}{\mathrm{d}T}h\right)_p = -T \cdot \left(\frac{\mathrm{d}^2}{\mathrm{d}T^2}f\right)_{V,\,dA/dV_T}$$

(in terms of the Helmholtz free energy, f)

If heat, Q, is slowly added to a material, allowing the material to expand reversibly at constant pressure, the heat added, Q, is related to the change in material temperature by the relation:

$$Q = \left(\int Cp\,\mathrm{d}T\right)_p = \Delta H$$

Functions that return Isobaric Specific Heat

$rp\_cptp(fluid, t, p)$	Obtains Specific isobaric Heat Capacity (C_p) of ***superheated vapor*** or ***compressed liquid*** in kJ/kg-K as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_cpft(fluid, t)$	Obtains Specific isobaric Heat Capacity (C_p) of ***saturated liquid*** in kJ/kg-K as a function of temperature (t) in K.
$rp\_cpfp(fluid, p)$	Obtains Specific isobaric Heat Capacity (C_p) of ***saturated liquid*** in kJ/kg-K as a function of Pressure (p) in MPa.
$rp\_cpgt(fluid, t)$	Obtains Specific isobaric Heat Capacity (C_p) of ***saturated vapor*** in kJ/kg-K as a function of temperature (t) in K.
$rp\_cpgp(fluid, p)$	Obtains Specific isobaric Heat Capacity (C_p) of ***saturated vapor*** in kJ/kg-K as a function of Pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$C_{ptp}(fluid, t, p) := \text{rp\_cptp}\left(fluid, \frac{t}{K}, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$$

$$C_{pft}(fluid, t) := \text{rp\_cpft}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg \cdot K} \qquad C_{pgt}(fluid, t) := \text{rp\_cpgt}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg \cdot K}$$

$$C_{pfp}(fluid, p) := \text{rp\_cpfp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg \cdot K} \qquad C_{pgp}(fluid, p) := \text{rp\_cpgp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$$

Example

$fluid :=$ "WATER.fld"

Temperatures

$n := 200$

$\Delta T := \left(T_c - T_t\right) \cdot \frac{1}{n}$

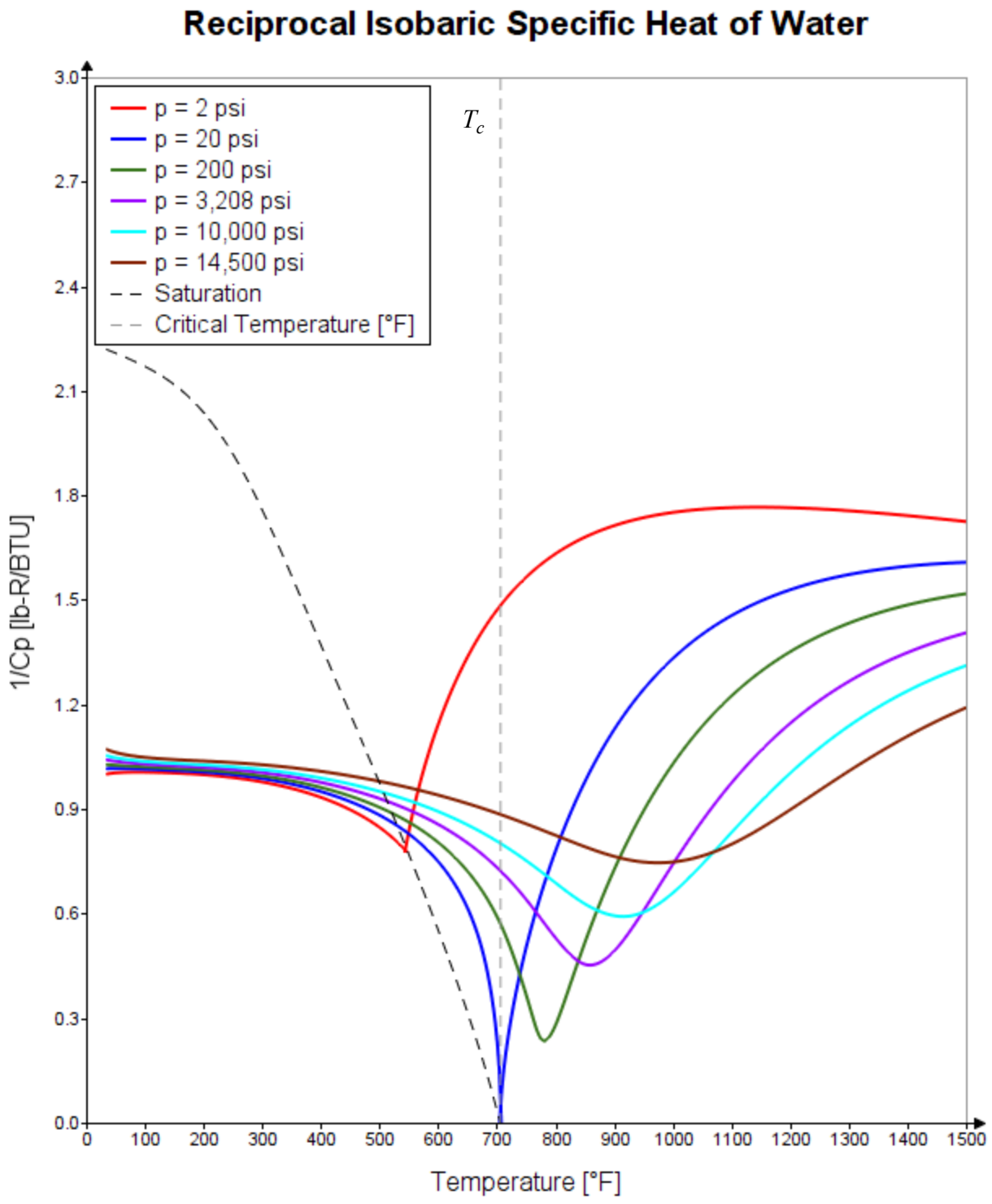
$t1 := T_t, T_t + \Delta T .. 2 \cdot T_c$

Pressures

$p1 := 1000 \cdot psi$ $\quad p4 := 7500 \cdot psi$

$p2 := 3200 \cdot psi$ $\quad p5 := 10000 \cdot psi$

$p3 := 5000 \cdot psi$ $\quad p6 := 14500 \cdot psi$


Reciprocal Isobaric Specific Heat of Water
p = 2 psi
p = 20 psi
p = 200 psi
p = 3,208 psi
p = 10,000 psi
p = 14,500 psi
Saturation
Critical Temperature [°F]
T_c
1/Cp [lb-R/BTU]
Temperature [°F]
3.0
2.7
2.4
2.1
1.8
1.5
1.2
0.9
0.6
0.3
0.0
0
100
200
300
400
500
600
700
800
900
1000
1100
1200
1300
1400
1500

A.6 Isochoric (Isometric) Specific Heat, Cv

Isochoric Specific Heat is defined as the rate of change of enthalpy of a unit mass of material with respect to temperature at a given constant volume (isochoric):

$$c_v = \left(\frac{\mathrm{d}}{\mathrm{d}T} u \right)_V$$

The expression above is evaluated using the internal energy, u. A similar expression can be derived in terms of the Helmholtz free energy.

If heat, Q, is slowly added to a material while keeping the volume constant, the heat added, Q, is related to the change in material temperature by the relation:

$$Q = \left(\int Cv \, \mathrm{d}T \right)_v = \Delta U$$

Functions that return Isobaric Specific Heat

$rp\_cvtp(fluid, t, p)$	Obtains Specific isochoric Heat Capacity (C_v) of ***superheated vapor*** or ***compressed liquid*** in kJ/kg-K as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_cvfp(fluid, p)$	Obtains Specific isochoric Heat Capacity (C_v) of ***saturated liquid*** in kJ/kg-K as a function of Pressure (p) in MPa.
$rp\_cvfp(fluid, p)$	Obtains Specific isochoric Heat Capacity (C_v) of ***saturated liquid*** in kJ/kg-K as a function of Pressure (p) in MPa.
$rp\_cvgp(fluid, p)$	Obtains Specific isochoric Heat Capacity (C_v) of ***saturated vapor*** in kJ/kg-K as a function of Pressure (p) in MPa.
$rp\_cvgp(fluid, p)$	Obtains Specific isochoric Heat Capacity (C_v) of ***saturated vapor*** in kJ/kg-K as a function of Pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$C_{vtp}(fluid, t, p) := \text{rp\_cvtp}\left(fluid, \frac{t}{K}, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$$

$$C_{vft}(fluid, t) := \text{rp\_cvft}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg \cdot K} \qquad C_{vgt}(fluid, t) := \text{rp\_cvgt}\left(fluid, \frac{t}{K}\right) \cdot \frac{kJ}{kg \cdot K}$$

$$C_{vfp}(fluid, p) := \text{rp\_cvfp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg \cdot K} \qquad C_{vgp}(fluid, p) := \text{rp\_cvgp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$$

Example

$fluid :=$ "fluids/WATER.fld"

Temperatures

$n := 200$

$\Delta T := (T_c - T_t) \cdot \frac{1}{n}$

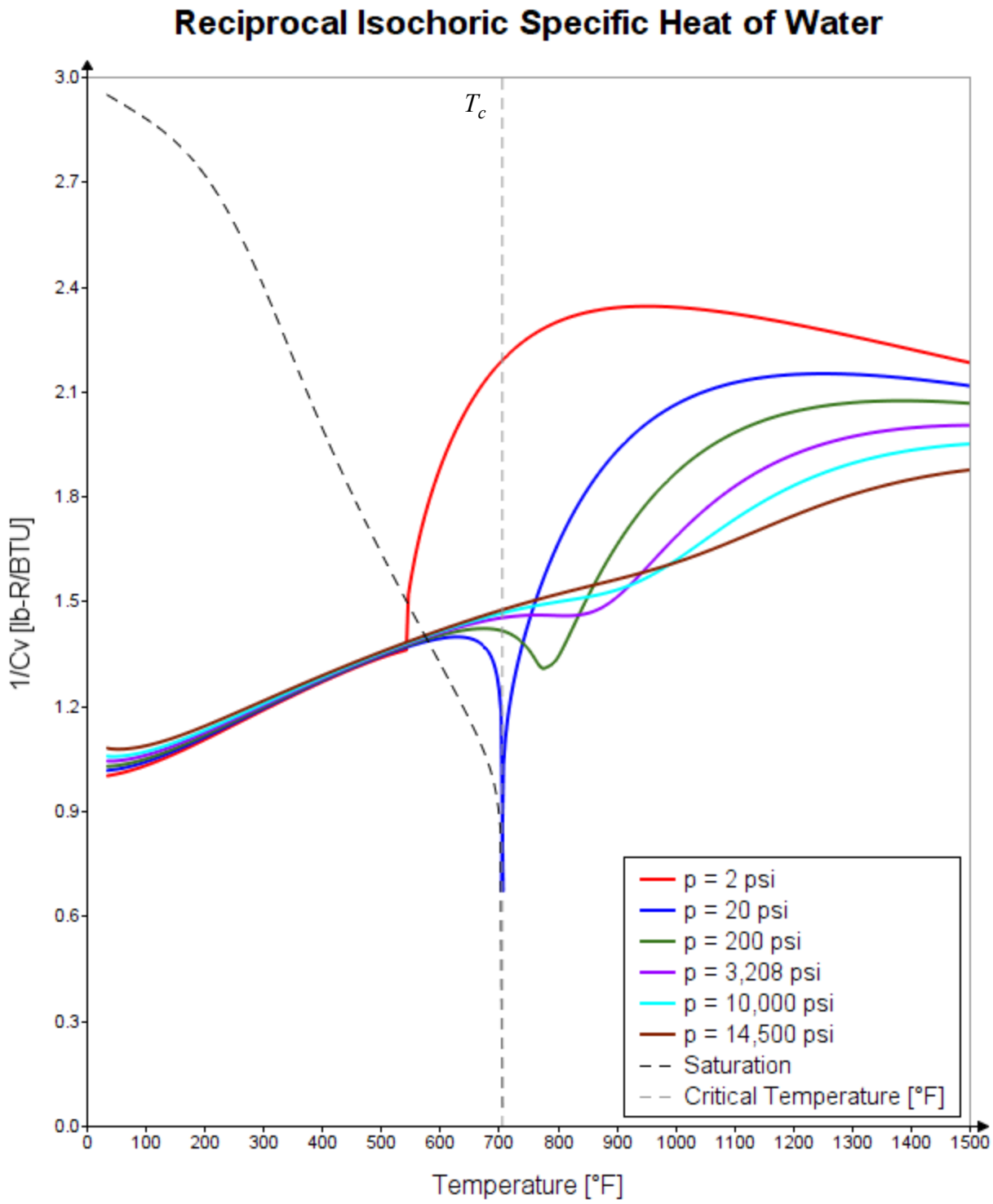
$t1 := T_t, T_t + \Delta T .. 2 \cdot T_c$

Pressures

$p1 := 1000 \cdot psi$ $p4 := 7500 \cdot psi$

$p2 := 3200 \cdot psi$ $p5 := 10000 \cdot psi$

$p3 := 5000 \cdot psi$ $p6 := 14500 \cdot psi$


Reciprocal Isochoric Specific Heat of Water
T_c
1/Cv [lb-R/BTU]
Temperature [°F]
3.0
2.7
2.4
2.1
1.8
1.5
1.2
0.9
0.6
0.3
0.0
0
100
200
300
400
500
600
700
800
900
1000
1100
1200
1300
1400
1500
p = 2 psi
p = 20 psi
p = 200 psi
p = 3,208 psi
p = 10,000 psi
p = 14,500 psi
Saturation
Critical Temperature [°F]

4.7 Speed of Sound, w

The speed of sound in a fluid is related to the specific volume and the fluid compressibility by the following thermodynamic relationship:

$$w^2 = -v^2 \cdot \left(\frac{\mathrm{d}}{\mathrm{d}v}P\right)_S = \left(\frac{\mathrm{d}}{\mathrm{d}\rho}P\right)_S$$

RefProp evaluates this differential using the Helmholtz free energy expression.

Functions that return Speed of Sound

$rp\_wtp(fluid, t, p)$	Obtains speed of sound through superheated vapor or compressed liquid in m/s as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_wft(fluid, t)$	Obtains speed of sound through ***saturated liquid*** in m/s as a function of Temperature (t) in K.
$rp\_wfp(fluid, p)$	Obtains speed of sound through ***saturated liquid*** in m/s as a function of Pressure (p) in MPa.
$rp\_wgt(fluid, t)$	Obtains speed of sound through ***saturated vapor*** in m/s as a function of Temperature (t) in K.
$rp\_wgp(fluid, p)$	Obtains speed of sound through ***saturated vapor*** in m/s as a function of Pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$w_{tp}(fluid, t, p) := \text{rp\_wtp}\left(fluid, \frac{t}{K}, \frac{p}{MPa}\right) \cdot \frac{m}{s}$$

$$w_{ft}(fluid, t) := \text{rp\_wft}\left(fluid, \frac{t}{K}\right) \cdot \frac{m}{s} \qquad w_{gt}(fluid, t) := \text{rp\_wgt}\left(fluid, \frac{t}{K}\right) \cdot \frac{m}{s}$$

$$w_{fp}(fluid, p) := \text{rp\_wfp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{m}{s} \qquad w_{gp}(fluid, p) := \text{rp\_wgp}\left(fluid, \frac{p}{MPa}\right) \cdot \frac{m}{s}$$

Example

$fluid := \text{“WATER.fld”}$

Temperatures (°F)

$n := 200$

$\Delta T := (T_c - T_t) \cdot \frac{1}{n}$

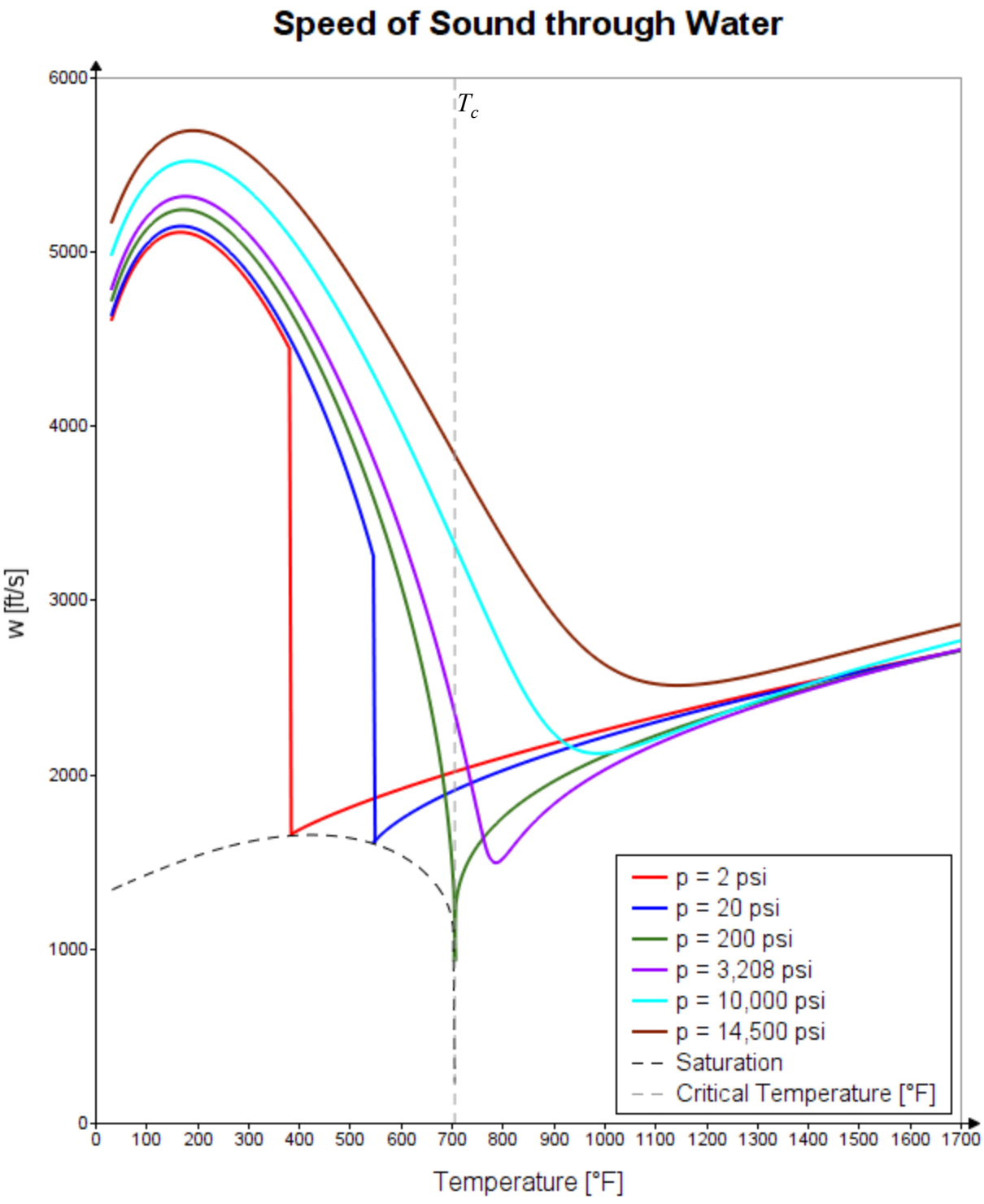
$T2 := T_t, T_t + \Delta T .. 2 \cdot T_c$

Pressures (psi)

$p1 := 200 \cdot psi$ $\quad p4 := 5000 \cdot psi$

$p2 := 1000 \cdot psi$ $\quad p5 := 10000 \cdot psi$

$p3 := 3200 \cdot psi$ $\quad p6 := 14500 \cdot psi$


Speed of Sound through Water
T_c
w [ft/s]
Temperature [°F]
0
1000
2000
3000
4000
5000
6000
0
100
200
300
400
500
600
700
800
900
1000
1100
1200
1300
1400
1500
1600
1700
p = 2 psi
p = 20 psi
p = 200 psi
p = 3,208 psi
p = 10,000 psi
p = 14,500 psi
Saturation
Critical Temperature [°F]

4.8 Saturation Curve

The ***Saturation Curve*** is defined as the Temperature / Pressure boundary curve that separates the liquid phase from the vapor phase. This curve starts at the ***Triple Point*** (where the solid, liquid, and vapor regions meet) to the ***Critical Point*** (where the fluid becomes super-critical; neither liquid or vapor).

The saturation curve and the temperature / pressure points at either end can all be retrieved from the RefProp functions for any fluid.

Functions that return Pressure

$rp\_ttrip(fluid, 0)$	Obtains the ***triple point*** temperature in K. Requires a dummy parameter (0).
$rp\_ptrip(fluid, 0)$	Obtains the ***triple point*** pressure in MPa. Requires a dummy parameter (0).
$rp\_tcrit(fluid, 0)$	Obtains the ***critical point*** temperature in K. Requires a dummy parameter (0).
$rp\_pcrit(fluid, 0)$	Obtains the ***critical point*** pressure in MPa. Requires a dummy parameter (0).
$rp\_tsatp(fluid, p)$	Obtains the ***saturation temperature*** in K from the given saturation pressure (p) in MPa. Valid from P_t to P_c.
$rp\_psatt(fluid, t)$	Obtains the ***saturation pressure*** in MPa from the given saturation temperature (t) in K. Valid from T_t to T_c.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$P_t(fluid) := \text{rp\_ptrip}(fluid, 0) \cdot MPa$ $\quad P_{sat}(fluid, t) := \text{rp\_psatt}\left(fluid, \frac{t}{K}\right) \cdot MPa$

$P_c(fluid) := \text{rp\_pcrit}(fluid, 0) \cdot MPa$

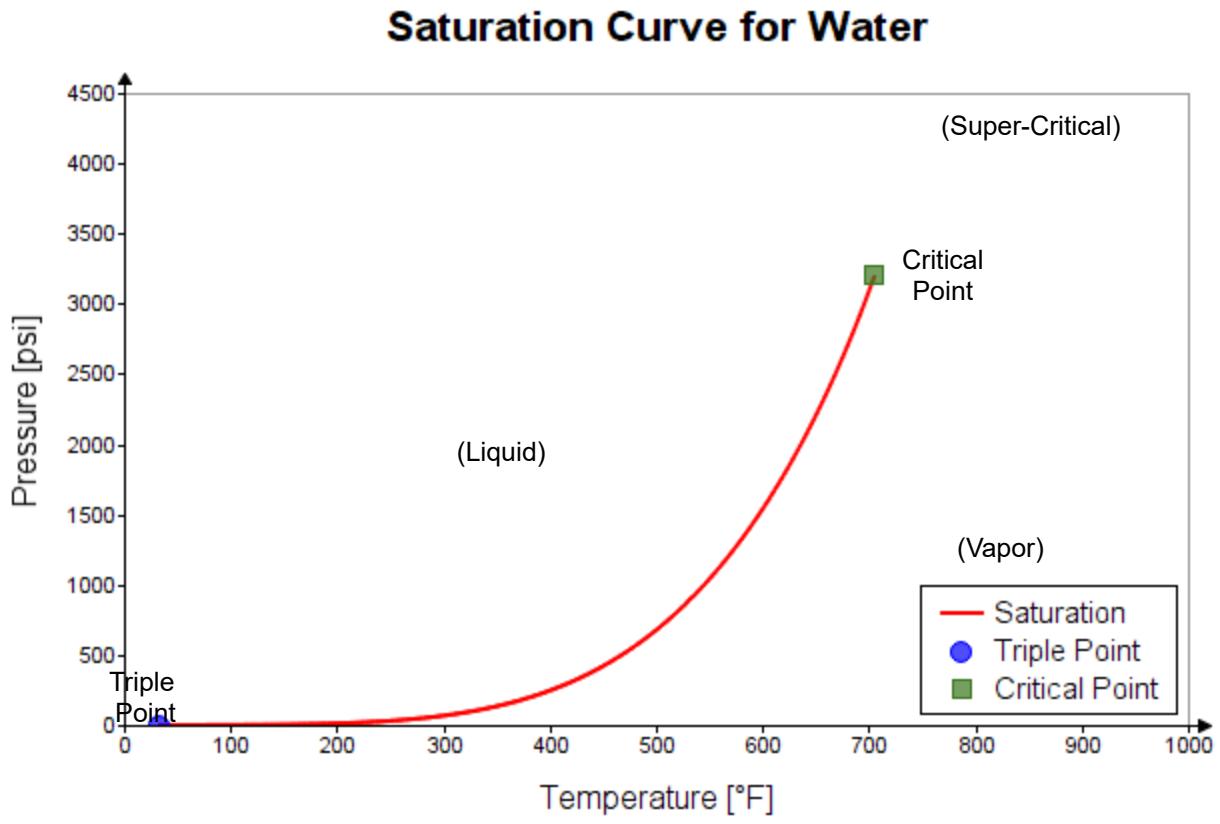
$T_t(fluid) := \text{rp\_ttrip}(fluid, 0) \cdot K$ $\quad T_{sat}(fluid, p) := \text{rp\_tsatp}\left(fluid, \frac{p}{MPa}\right) \cdot K$

$T_c(fluid) := \text{rp\_tcrit}(fluid, 0) \cdot K$

Example

$fl := \text{“WATER.fld”}$ $\quad n := 1000$

Temperatures $\quad \Delta T := \left(T_c(fl) - T_t(fl)\right) \cdot \frac{1}{n}$ $\quad TI := T_t(fl), T_t(fl) + \Delta T .. T_c(fl)$



4.9 Surface Tension, σ

Surface Tension is a property that describes the work required to create a unit surface area of liquid water. Its units are N-m/m² or N/m and it is a function of temperature alone (decreases with increasing temperature). Surface Tension is important in the formation of bubbles and in the study of atomization. As this is a property of the vapor / liquid interface, it is only defined along the saturation curve and has no meaning in the single phase regions.

Functions that return Surface Tension

$rp\_surften(fluid, t)$ Obtains surface tension of the fluid in kg/s² as a function of temperature (t) in K.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Example

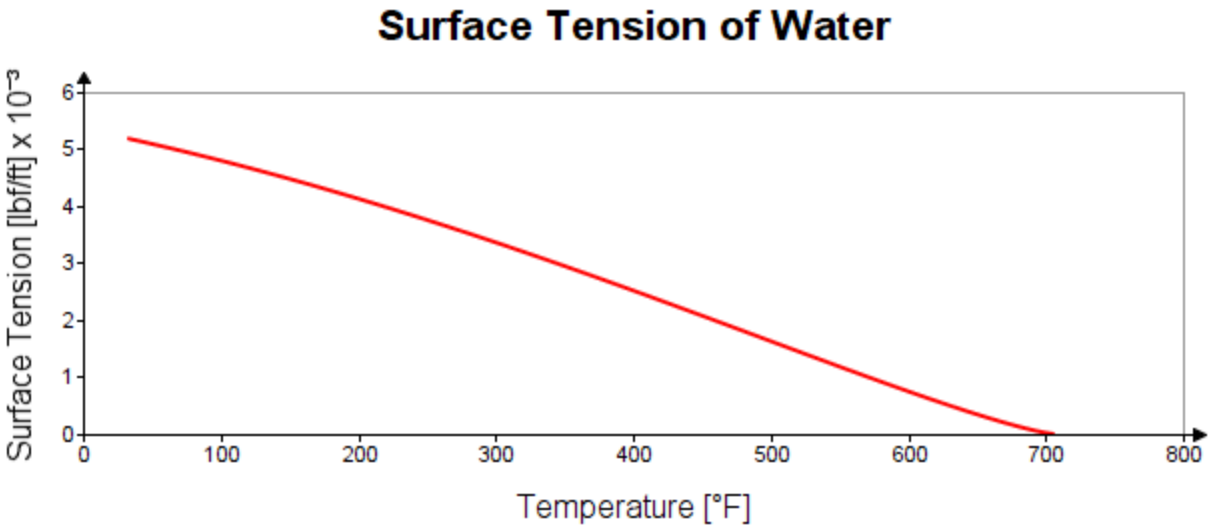
$fluid := \text{“WATER.fld”}$ $n := 100$

Temperatures (°F)

$\Delta T_s := (T_c(fluid) - T_t(fluid)) \div n$

$T_s := T_t(fluid), T_t(fluid) + \Delta T_s .. T_c(fluid)$

$\sigma(T) := \text{rp\_surften}\left(fluid, \frac{T}{K}\right) \cdot \frac{kg}{s^2}$



4.10 Thermal Conductivity, k

Thermal Conductivity arises from the law of conduction in the study of heat transfer. It is often referred to as a constant of proportionality. It provides important information on the heat conduction rate given a temperature gradient and surface area perpendicular to the heat flow. Thermal conductivity is also used in many empirically derived relationships which are important in the study of heat transfer.

Functions that return Thermal Conductivity

$rp\_ktp(fluid, t, p)$	Obtains thermal conductivity (k) of superheated vapor or compressed liquid in kW/m-K as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_kft(fluid, t)$	Obtains thermal conductivity (k) of saturated liquid in kW/m-K as a function of Temperature (t) in K.
$rp\_kfp(fluid, p)$	Obtains thermal conductivity (k) of saturated liquid in kW/m-K as a function of Pressure (p) in MPa.
$rp\_kgt(fluid, t)$	Obtains thermal conductivity (k) of saturated vapor in kW/m-K as a function of Temperature (t) in K.
$rp\_kgp(fluid, p)$	Obtains thermal conductivity (k) of saturated vapor in kW/m-K as a function of Pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$k_{tp}(fluid, T, P) := \text{rp\_ktp}\left(fluid, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{W}{m \cdot K}$$

$$k_{ft}(fluid, T) := \text{rp\_kft}\left(fluid, \frac{T}{K}\right) \cdot \frac{W}{m \cdot K} \qquad k_{gt}(fluid, T) := \text{rp\_kgt}\left(fluid, \frac{T}{K}\right) \cdot \frac{W}{m \cdot K}$$

$$k_{fp}(fluid, P) := \text{rp\_kfp}\left(fluid, \frac{P}{MPa}\right) \cdot \frac{W}{m \cdot K} \qquad k_{gp}(fluid, P) := \text{rp\_kgp}\left(fluid, \frac{P}{MPa}\right) \cdot \frac{W}{m \cdot K}$$

Example $fluid := \text{“WATER.fld”}$

Pressures (psi)

$p1 := 200 \cdot psi$ $p4 := 5000 \cdot psi$

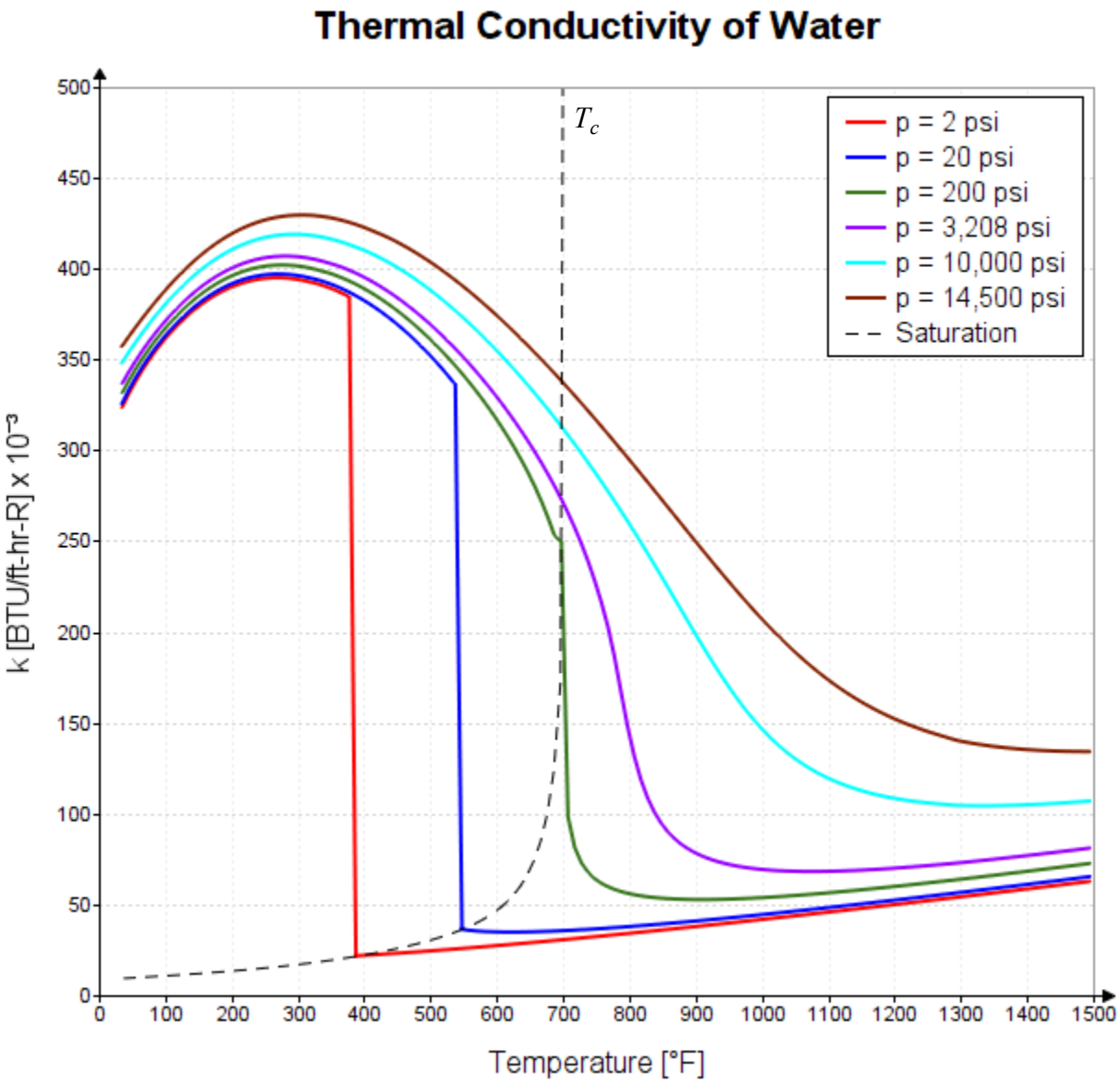
$p2 := 1000 \cdot psi$ $p5 := 10000 \cdot psi$

$p3 := 3000 \cdot psi$ $p6 := 14500 \cdot psi$

Temperatures (°F)

$t1 := 35\ °F, 45\ °F .. 1500\ °F$

$T_c := \text{rp\_tcrit}(fluid, 0) \cdot K$



4.11 Viscosity, μ

Viscosity is a property derived out of the study of Newtonian fluid mechanics. To analyze fluid motion, it is necessary to express the resistive forces in terms of velocity. The resistance to the shearing motion of flow is given by viscosity. It is used in determining dimensionless parameters (i.e. Reynold's number, etc.) as well as in many empirical correlations for heat transfer and fluid flow.

Functions that return Viscosity

$rp\_mutp(fluid, t, p)$	Obtains absolute (dynamic) viscosity of superheated vapor or compressed liquid in μPa-sec as a function of temperature (t) in K and Pressure (p) in MPa.
$rp\_muft(fluid, t)$	Obtains absolute (dynamic) viscosity of saturated liquid in μPa-sec as a function of Temperature (t) in K.
$rp\_mufp(fluid, p)$	Obtains absolute (dynamic) viscosity of saturated liquid in μPa-sec as a function of Pressure (p) in MPa.
$rp\_mugt(fluid, t)$	Obtains absolute (dynamic) viscosity of saturated vapor in μPa-sec as a function of Temperature (t) in K.
$rp\_mugp(fluid, p)$	Obtains absolute (dynamic) viscosity of saturated vapor in μPa-sec as a function of Pressure (p) in MPa.

NOTE : The first parameter in all of these functions is a **"fluid string"** variable that identifies the fluid material to be used for the property calculations.

Application of Units

$$\mu_{tp}(fluid, T, P) := \text{rp\_mutp}\left(fluid, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \mu Pa \cdot s$$

$$\mu_{ft}(fluid, T) := \text{rp\_muft}\left(fluid, \frac{T}{K}\right) \cdot \mu Pa \cdot s \qquad \mu_{gt}(fluid, T) := \text{rp\_mugt}\left(fluid, \frac{T}{K}\right) \cdot \mu Pa \cdot s$$

$$\mu_{fp}(fluid, P) := \text{rp\_mufp}\left(fluid, \frac{P}{MPa}\right) \cdot \mu Pa \cdot s \qquad \mu_{gp}(fluid, P) := \text{rp\_mugp}\left(fluid, \frac{P}{MPa}\right) \cdot \mu Pa \cdot s$$

Example

$fluid := \text{“WATER.fld”}$

Pressures

$p1 := 200 \cdot psi$ $p4 := 5000 \cdot psi$

$p2 := 1000 \cdot psi$ $p5 := 10000 \cdot psi$

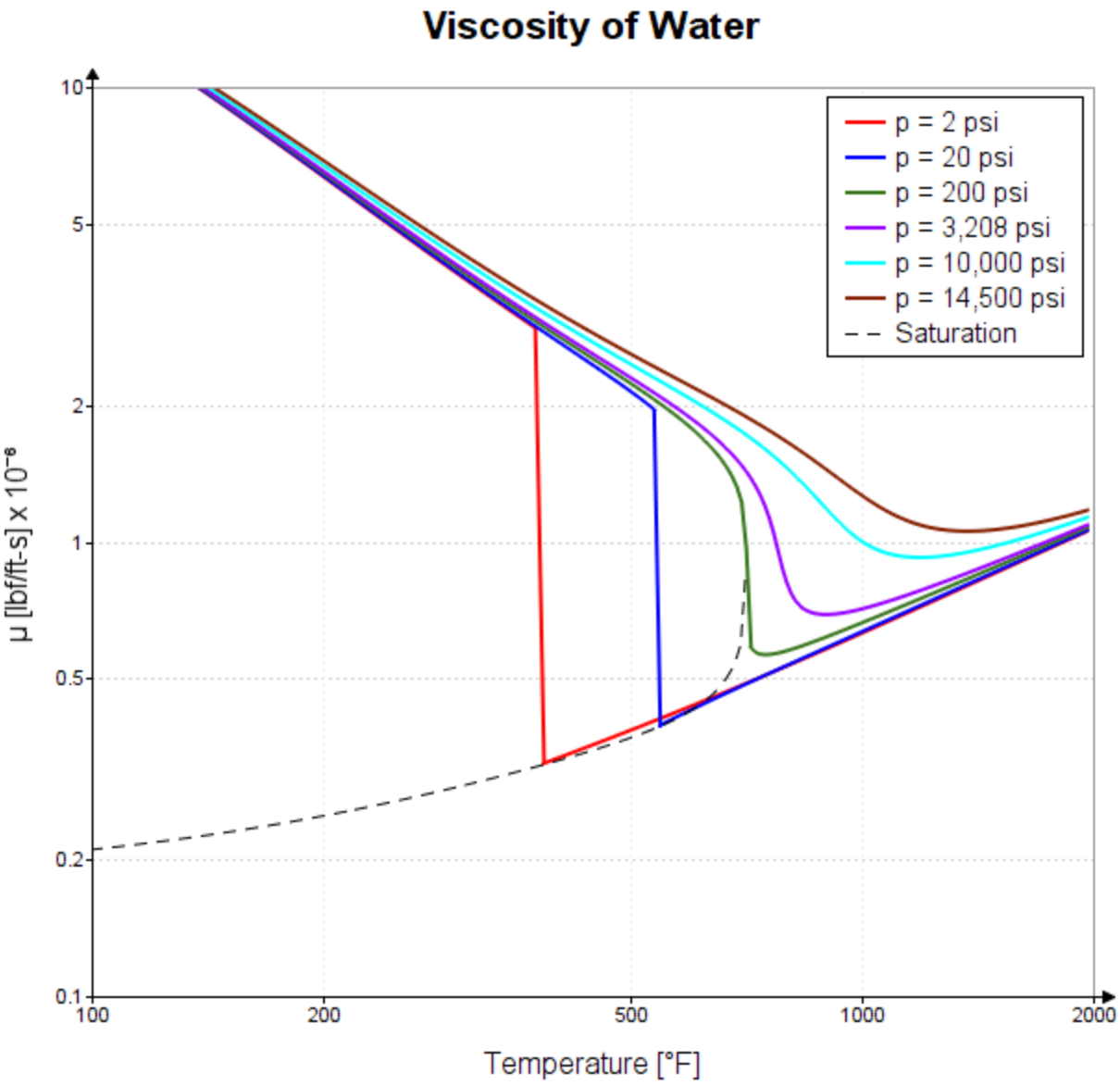
$p3 := 3200 \cdot psi$ $p6 := 14500 \cdot psi$

Temperatures

$t1 := 35\ °F, 45\ °F .. 5000\ °F$

$T_c := \text{rp\_tcrit}(fluid, 0) \cdot K$

$T_c = 705.103\ °F$



Appendix B - RefProp Function Verification

The ***RefProp*** functions have been tested and verified against other RefProp implementations and published data. Verification is based on the following criteria.

Continuity Check

All supplied functions have been plotted over the valid property limits for all of the fluids in the NIST RefProp Library to ensure continuity of the data with no areas of non-convergence of the underlying NIST RefProp functions. Water, CO2, and most of the room temperature gasses have been shown to be continuous over the full Temperature and Pressure limits of the models. This includes ranges tightly focused around the critical point (Figure 1).

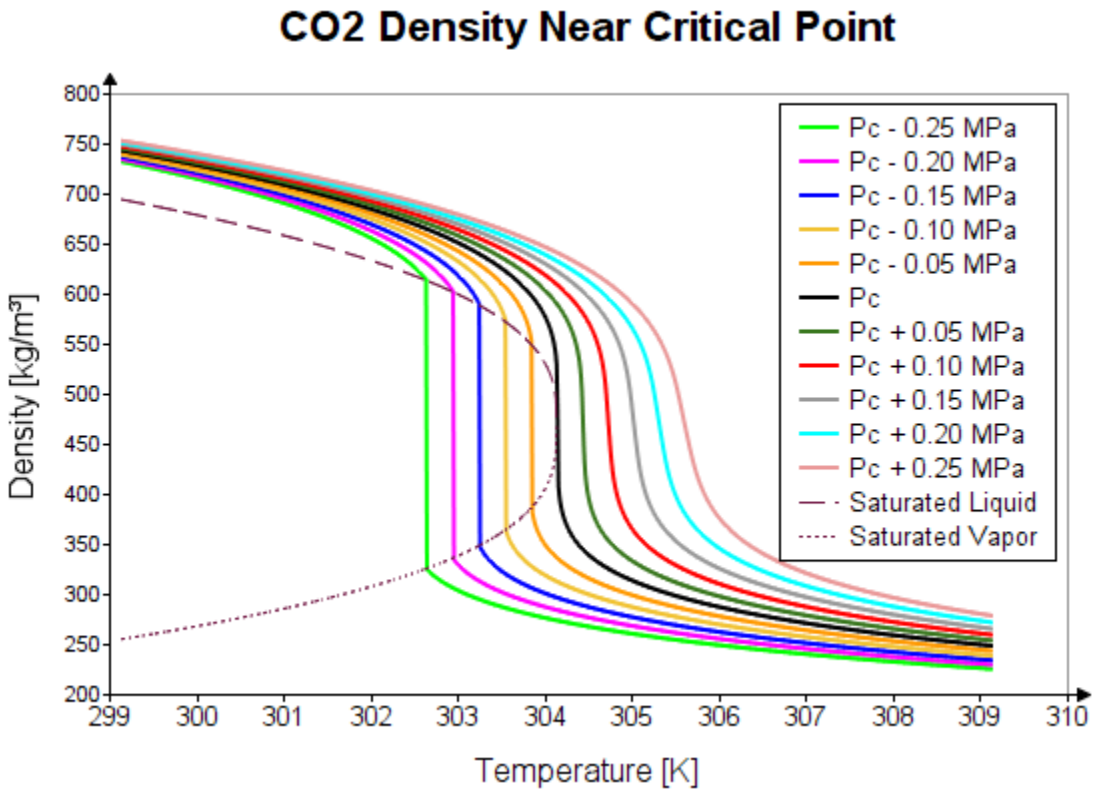


Figure 1 : Continuity of Calculated Density of Carbon Dioxide in the Vicinity of the Critical Point

When using any fluid from RefProp (or any properties library), care should be taken to verify the continuity over the region of interest, and accuracy of the saturation curve as it approaches the critical point; similar to Figure 1 and Figure 2.

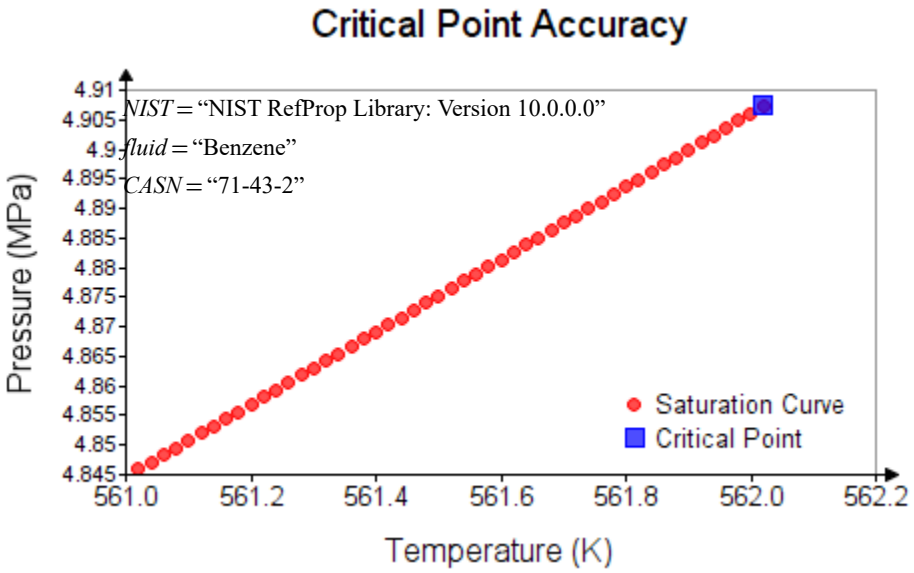


Figure 2 : Continuity of Benzene Saturation Curve up to the Critical Point

Accuracy

Comparisons have been made for CO_2 property results from the Mathcad DLL and the NIST provided Excel Add-in at selected Temperature and Pressure point locations. These comparisons (Figure 3) show that the two implementations agree to 5 significant figures or better on the calculated thermodynamic and transport properties. This comparison provides verification that the Mathcad Add-In is at least as accurate as the NIST Library, but makes no claim for the accuracy of the NIST Library itself.

Error =

	ρ	H	S	U	Cp	Cv	w	μ	k
	1	2	3	4	5	6	7	8	9
1	$4.3 \cdot 10^{6}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$9.9 \cdot 10^{8}$	$1.3 \cdot 10^{6}$	$-1.5 \cdot 10^{5}$
2	$4.3 \cdot 10^{6}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$5.1 \cdot 10^{8}$	$1.3 \cdot 10^{6}$	$-1.5 \cdot 10^{5}$
3	$5.3 \cdot 10^{6}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.4 \cdot 10^{5}$	$-1.5 \cdot 10^{5}$	$-2.5 \cdot 10^{7}$	$1.9 \cdot 10^{6}$	$-1.4 \cdot 10^{5}$
4	$1.3 \cdot 10^{5}$	$-2 \cdot 10^{5}$	$-1.9 \cdot 10^{5}$	$-2 \cdot 10^{5}$	$-1.2 \cdot 10^{4}$	$-3.1 \cdot 10^{5}$	$1.9 \cdot 10^{5}$	$1.5 \cdot 10^{5}$	$-3.2 \cdot 10^{5}$
5	$3.3 \cdot 10^{6}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.7 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$1.4 \cdot 10^{6}$	$2.4 \cdot 10^{6}$	$-1.4 \cdot 10^{5}$
6	$4.4 \cdot 10^{6}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$-1.6 \cdot 10^{5}$	$1.4 \cdot 10^{7}$	$1.5 \cdot 10^{6}$	$-1.5 \cdot 10^{5}$

$$\xrightarrow{\text{min\_sig\_digits (Error)}} = \begin{pmatrix} 6 & 5 & 5 & 5 & 5 & 5 & 8 & 6 & 5 \\ 6 & 5 & 5 & 5 & 5 & 5 & 8 & 6 & 5 \\ 6 & 5 & 5 & 5 & 5 & 5 & 7 & 6 & 5 \\ 5 & 5 & 5 & 5 & 4 & 5 & 5 & 5 & 5 \\ 6 & 5 & 5 & 5 & 5 & 5 & 6 & 6 & 5 \\ 6 & 5 & 5 & 5 & 5 & 5 & 7 & 6 & 5 \end{pmatrix}$$

$$RMS_{error} = 2.2 \times 10^{-5}$$

$$\text{precision} = 5$$

The Mathcad and Excel implementations of the NIST Refprop DLL agree to within 5 significant figures over the comparison points chosen.

Figure 3 : Mathcad Add-In Accuracy Relative to the NIST MS-Excel Add-In

Since the RefProp property models for Water are taken from the IAPWS-95 Steam/ Water Property Formulation for General and Scientific Use, the Mathcad predicted water properties were compared directly with the IAPWS-95 published verification values. Thus providing verification that the Mathcad Add-In implementation functioning as expected. The IAPWS-95 release document provides a table of property values over a matrix of state points (Figure 4) lying in all liquid/vapor/super-critical phase regions of the phase diagram.

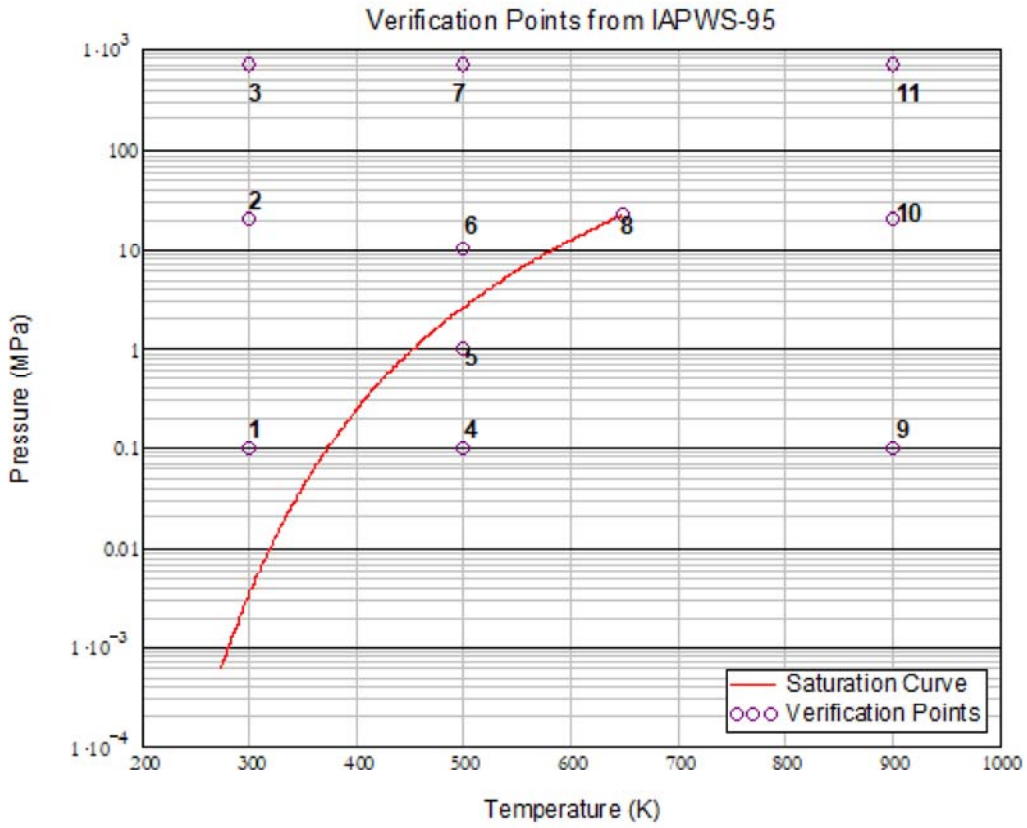


Figure 4 : IAPWS-95 Recommended Verification Points

Using the direct functions of Temperature and Density, the predicted pressures agree with the published verification values to 6 significant figures or better. All other properties (C_v, w, and S) agreed to 9 significant figures or better, which is the precision of the published values.

Indirectly using the functions of Temperature and Pressure to calculate Density, requires an iterative procedure within the NIST RefProp Library that solves the Pressure equation for Density to a selected tolerance. This function evaluates Density to ***6 significant figures*** (or better) of the published values. Since this Density calculation is used as a first step in the calculation of all other properties as a function of Temperature and Pressure, the remaining properties are also limited to an accuracy of ***6 significant figures***. According to the IAPWS-95 documentation, the formulation itself (and hence the verification values) is only accurate to 5 significant figures or less in the liquid region, and 4 significant figures or less in the vapor and super-critical regions.

Appendix C Unit Functions Reference

NIST RefProp Functions with Appropriately Applied Units

(For Mathcad Prime and Version 2.0 or later of the RefProp Add-In)

Most property routines listed below require a fluid string, fl, to specify the fluid properties to use. Make calls to $rp\_property$ functions with one of the RefProp fluids (e.g. $fl := \text{“CO2.fld”}$) or mixtures. Fluid names are not case sensitive.

Function to get the version string:

$Refprop\_Addin\_Version(i) := rp\_getvers(i)$

$NIST\_Refprop\_Version(i) := rp\_getNIST(i)$

Fluid Information Functions

$Name(fl, component) := rp\_getname(fl, component)$ Returns a string with the full fluid name

$CASN(fl, component) := rp\_getcasn(fl, component)$ Returns a string containing the CAS number

The following functions all retrieve specific properties and constants. The parameters to each function (unless it's a component number) should have appropriate Mathcad units applied. Each of these functions will convert the passed parameters to the units expected by the RefProp functions, and return values with appropriate units applied. Temperatures should always be passed with absolute temperature units of K or R.

Basic Physical Constants by individual component (component = 1 for a pure fluid)

Molecular Weight $WM(fl, comp) := rp\_wmol(fl, comp) \cdot \frac{gm}{mol}$

$R_g(fl, comp) := rp\_rgas(fl, comp) \cdot J \cdot mol^{-1} \cdot K^{-1}$

	Critical Point	Triple Point
Temperature	$T_c(fl, component) := rp\_tcrit(fl, component) \cdot K$	$T_t(fl, component) := rp\_ttrip(fl, component) \cdot K$
Pressure	$P_c(fl, component) := rp\_pcrit(fl, component) \cdot MPa$	$P_t(fl, component) := rp\_ptrip(fl, component) \cdot MPa$
Density	$\rho_c(fl, component) := rp\_rhocrit(fl, component) \cdot \frac{kg}{m^3}$	

Temperature and Pressure Functions along the Saturation Curve

$T_{sat}(fl, P) := \text{rp\_tsatp}\left(fl, \frac{P}{MPa}\right) \cdot K$

$P_{sat}(fl, T) := rp\_psatt\left(fl, \frac{T}{K}\right) \cdot MPa$

Thermodynamic Properties as functions of Temperature and Pressure

Fluid function subscripts indicate the phase region and the independent parameter to be used.
A subscript of **f** indicates saturated liquid (fluid) and a subscript of **g** indicates saturated vapor (*gas*). If neither **f** or **g** is used, the function returns either subcooled liquid or superheated vapor values. These functions are only defined between the triple point and the critical point, and will return an error otherwise.

A second subscript of **t**, **p**, or both indicates the independent parameter that should be passed. Thus, ρ_{fp} would return the saturated liquid density as a function of the saturation pressure, p. Alternatively, ρ_{tp} will return the general subcooled liquid or superheated vapor density as a function of both temperature, t, and pressure, p.

Density, ρ

Saturated Liquid as a function of temperature: $\rho_{fi}(fl, T) := \text{rp\_rhoft}\left(fl, \frac{T}{K}\right) \cdot \frac{kg}{m^3}$

Saturated Vapor as a function of temperature: $\rho_{gt}(fl, T) := \text{rp\_rhogt}\left(fl, \frac{T}{K}\right) \cdot \frac{kg}{m^3}$

Saturated Liquid as a function of temperature: $\rho_{fp}(fl, P) := \text{rp\_rhofp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kg}{m^3}$

Saturated Vapor as a function of temperature: $\rho_{gp}(fl, P) := \text{rp\_rhogp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kg}{m^3}$

Subcooled Liquid or Superheated Vapor as a function of both temperature and pressure: $\rho_{tp}(fl, T, P) := \text{rp\_rhotp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kg}{m^3}$

Enthalpy, **h**

$h_{ft}(fl, T) := \text{rp\_hft}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg}$

$h_{gt}(fl, T) := \text{rp\_hgt}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg}$

$h_{fp}(fl, P) := \text{rp\_hfp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg}$

$h_{gp}(fl, P) := \text{rp\_hgp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg}$

$h_{tp}(fl, T, P) := \text{rp\_htp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg}$

Entropy, **s**

$s_{ft}(fl, T) := \text{rp\_sft}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg \cdot K}$

$s_{gt}(fl, T) := \text{rp\_sgt}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg \cdot K}$

$s_{fp}(fl, P) := \text{rp\_sfp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

$s_{gp}(fl, P) := \text{rp\_sgp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

$s_{tp}(fl, T, P) := \text{rp\_stp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

Internal Energy, **u**

$u_{fp}(fl, P) := \text{rp\_ufp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg}$

$u_{gp}(fl, P) := \text{rp\_ugp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg}$

$u_{ft}(fl, T) := \text{rp\_uft}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg}$

$u_{gt}(fl, T) := \text{rp\_ugt}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg}$

$u_{tp}(fl, T, P) := \text{rp\_utp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg}$

Isobaric Specific Heat, **Cp**

$Cp_{fp}(fl, P) := \text{rp\_cpfp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cp_{gp}(fl, P) := \text{rp\_cpgp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cp_{ft}(fl, T) := \text{rp\_cpft}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cp_{gt}(fl, T) := \text{rp\_cpgt}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cp_{tp}(fl, T, P) := \text{rp\_cptp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

Isochoric Specific Heat, **Cv**

$Cv_{fp}(fl, P) := \text{rp\_cvfp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cv_{gp}(fl, P) := \text{rp\_cvgp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cv_{ft}(fl, T) := \text{rp\_cvft}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cv_{gt}(fl, T) := \text{rp\_cvgt}\left(fl, \frac{T}{K}\right) \cdot \frac{kJ}{kg \cdot K}$

$Cv_{tp}(fl, T, P) := \text{rp\_cvtp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{kJ}{kg \cdot K}$

Sonic Velocity, w

$w_{fp}(fl, P) := \text{rp\_wfp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{m}{s}$

$w_{gp}(fl, P) := \text{rp\_wgp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{m}{s}$

$w_{ft}(fl, T) := \text{rp\_wft}\left(fl, \frac{T}{K}\right) \cdot \frac{m}{s}$

$w_{gt}(fl, T) := \text{rp\_wgt}\left(fl, \frac{T}{K}\right) \cdot \frac{m}{s}$

$w_{tp}(fl, T, P) := \text{rp\_wtp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{m}{s}$

Transport Properties as Functions of Temperature and Pressure

Thermal Conductivity, **k**

$k_{fp}(fl, P) := \text{rp\_kfp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{W}{m \cdot K}$

$k_{gp}(fl, P) := \text{rp\_kgp}\left(fl, \frac{P}{MPa}\right) \cdot \frac{W}{m \cdot K}$

$k_{ft}(fl, T) := \text{rp\_kft}\left(fl, \frac{T}{K}\right) \cdot \frac{W}{m \cdot K}$

$k_{gt}(fl, T) := \text{rp\_kgt}\left(fl, \frac{T}{K}\right) \cdot \frac{W}{m \cdot K}$

$k_{tp}(fl, T, P) := \text{rp\_ktp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot \frac{W}{m \cdot K}$

Fluid Viscosity, **μ**

$\mu_{fp}(fl, P) := \text{rp\_mufp}\left(fl, \frac{P}{MPa}\right) \cdot \mu Pa \cdot s$

$\mu_{gp}(fl, P) := \text{rp\_mugp}\left(fl, \frac{P}{MPa}\right) \cdot (\mu Pa \cdot s)$

$\mu_{ft}(fl, T) := \text{rp\_muft}\left(fl, \frac{T}{K}\right) \cdot (\mu Pa \cdot s)$

$\mu_{gt}(fl, T) := \text{rp\_mugt}\left(fl, \frac{T}{K}\right) \cdot (\mu Pa \cdot s)$

$\mu_{tp}(fl, T, P) := \text{rp\_mutp}\left(fl, \frac{T}{K}, \frac{P}{MPa}\right) \cdot (\mu Pa \cdot s)$

Some Additional Property Functions

Surface Tension

$$\sigma_t(fl, T) := \text{rp\_surften}\left(fl, \frac{T}{K}\right) \cdot \frac{N}{m}$$

Derived Quantities

Coefficient of Thermal Expansion

$$\beta(fl, T, P) := \frac{-1}{\rho_{tp}(fl, T, P)} \cdot \left(\frac{\mathrm{d}}{\mathrm{d}T}\rho_{tp}(fl, T, P)\right)$$

Isentropic Exponent

$$\gamma(fl, T, P) := \frac{Cp_{tp}(fl, T, P)}{Cv_{tp}(fl, T, P)}$$

Latent Heat

$$h_{fg}(fl, t) := h_{gt}(fl, t) - h_{ft}(fl, t)$$

Vapor Quality

$$x_{th}(f, t, h) := min\left(\max\left(\frac{h - h_{ft}(f, t)}{h_{gt}(f, t) - h_{ft}(f, t)}, 0\right), 1.0\right)$$

$$x_{ts}(f, t, s) := min\left(\max\left(\frac{s - s_{ft}(f, t)}{s_{gt}(f, t) - s_{ft}(f, t)}, 0\right), 1.0\right)$$

$$x_{t\rho}(f, t, \rho) := min\left(max\left(\frac{\rho - \rho_{ft}(f, t)}{\rho_{gt}(f, t) - \rho_{ft}(f, t)}, 0\right), 1.0\right)$$

$$x_{tu}(f, t, u) := min\left(\max\left(\frac{u - u_{ft}(f, t)}{u_{gt}(f, t) - u_{ft}(f, t)}, 0\right), 1.0\right)$$

2-phase Properties

$$h_{tx}(f, t, x) := h_{ft}(f, t) \cdot (1 - x) + h_{gt}(f, t) \cdot x$$

$$s_{tx}(f, t, x) := s_{ft}(f, t) \cdot (1 - x) + s_{gt}(f, t) \cdot x$$

$$u_{tx}(f, t, x) := u_{ft}(f, t) \cdot (1 - x) + u_{gt}(f, t) \cdot x$$

$$\rho_{tx}(f, t, x) := \left(\rho_{ft}(f, t)^{-1} \cdot (1 - x) + \rho_{gt}(f, t)^{-1} \cdot x\right)^{-1}$$

$$\Delta\rho_t(f, t) := \rho_{ft}(f, t) - \rho_{gt}(f, t)$$

Compressibility

$$Z_{tp}(fl, T, P) := \frac{P \cdot WM(fl, 0)}{\rho_{tp}(fl, T, P) \cdot R_g(fl, 0) \cdot T}$$

$$Z_{ft}(fl, T) := \frac{P_{sat}(fl, T) \cdot WM(fl, 0)}{\rho_{ft}(fl, T) \cdot R_g(fl, 0) \cdot T}$$

$$Z_{gt}(fl, T) := \frac{P_{sat}(fl, T) \cdot WM(fl, 0)}{\rho_{gt}(fl, T) \cdot R_g(fl, 0) \cdot T}$$

$$Z_{fp}(fl, P) := \frac{P \cdot WM(fl, 0)}{\rho_{fp}(fl, P) \cdot R_g(fl, 0) \cdot T_{sat}(fl, P)}$$

$$Z_{gp}(fl, P) := \frac{P \cdot WM(fl, 0)}{\rho_{gp}(fl, P) \cdot R_g(fl, 0) \cdot T_{sat}(fl, P)}$$

$$Z_c(fl, comp) := \frac{P_c(fl, comp) \cdot WM(fl, comp)}{\rho_c(fl, comp) \cdot R_g(fl, comp) \cdot T_c(fl, comp)}$$

$$x_{ph}(f, p, h) := min\left(\max\left(\frac{h - h_{fp}(f, p)}{h_{gp}(f, p) - h_{fp}(f, p)}, 0\right), 1.0\right)$$

$$x_{ps}(f, p, s) := min\left(\max\left(\frac{s - s_{fp}(f, p)}{s_{gp}(f, p) - s_{fp}(f, p)}, 0\right), 1.0\right)$$

$$x_{p\rho}(f, p, \rho) := min\left(max\left(\frac{\rho - \rho_{fp}(f, p)}{\rho_{gp}(f, p) - \rho_{fp}(f, p)}, 0\right), 1.0\right)$$

$$x_{pu}(f, p, u) := min\left(\max\left(\frac{u - u_{fp}(f, p)}{u_{gp}(f, p) - u_{fp}(f, p)}, 0\right), 1.0\right)$$

$$h_{px}(f, p, x) := h_{fp}(f, p) \cdot (1 - x) + h_{gp}(f, p) \cdot x$$

$$s_{px}(f, p, x) := s_{fp}(f, p) \cdot (1 - x) + s_{gp}(f, p) \cdot x$$

$$u_{px}(f, p, x) := u_{fp}(f, p) \cdot (1 - x) + u_{gp}(f, p) \cdot x$$

$$\rho_{px}(f, p, x) := \left(\rho_{fp}(f, p)^{-1} \cdot (1 - x) + \rho_{gp}(f, p)^{-1} \cdot x\right)^{-1}$$

$$\Delta\rho_p(f, p) := \rho_{fp}(f, p) - \rho_{gp}(f, p)$$

Morton Number

$$Mo(f,P) := g \cdot \mu_{fp}(f,P)^4 \cdot \frac{\Delta\rho_p(f,P)}{\rho_{fp}(f,P)^2 \cdot \sigma_t(f, T_{sat}(f,P))^3}$$

Dynamic Viscosity

$$v_{fp}(f,P) := \frac{\mu_{fp}(f,P)}{\rho_{fp}(f,P)} \qquad v_{gp}(f,P) := \frac{\mu_{gp}(f,P)}{\rho_{gp}(f,P)}$$

$$v_{ft}(f,T) := \frac{\mu_{ft}(f,T)}{\rho_{ft}(f,T)} \qquad v_{gt}(f,T) := \frac{\mu_{gt}(f,T)}{\rho_{gt}(f,T)}$$

$$v_{tp}(f,T,P) := \frac{\mu_{tp}(f,T,P)}{\rho_{tp}(f,T,P)}$$

Prandtl Number

$$Pr_{fp}(f,P) := \mu_{fp}(f,P) \cdot Cp_{fp}(f,P) \cdot k_{fp}(f,P)^{-1} \qquad Pr_{gp}(f,P) := \mu_{gp}(f,P) \cdot Cp_{gp}(f,P) \cdot k_{gp}(f,P)^{-1}$$

$$Pr_{ft}(f,T) := \mu_{ft}(f,T) \cdot Cp_{ft}(f,T) \cdot k_{ft}(f,T)^{-1} \qquad Pr_{gt}(f,T) := \mu_{gt}(f,T) \cdot Cp_{gt}(f,T) \cdot k_{gt}(f,T)^{-1}$$

$$Pr_{tp}(f,T,P) := \mu_{tp}(f,T,P) \cdot Cp_{tp}(f,T,P) \cdot k_{tp}(f,T,P)^{-1}$$

Reverse Functions of Pressure/Temperature and Enthalpy/Entropy

$$T_{ph}(fl, p, h) := \text{rp\_tph}\left(fl, \frac{p}{MPa}, h \cdot \frac{kg}{kJ}\right) \cdot K$$

$$T_{hs}(fl, h, s) := \text{rp\_ths}\left(fl, h \cdot \frac{kg}{kJ}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot K$$

$$T_{ps}(fl, p, s) := \text{rp\_tps}\left(fl, \frac{p}{MPa}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot K$$

$$P_{hs}(fl, h, s) := \text{rp\_phs}\left(fl, h \cdot \frac{kg}{kJ}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot MPa$$

$$P_{th}(fl, t, h, r) := \text{rp\_pth}\left(fl, \frac{t}{K}, h \cdot \frac{kg}{kJ}, r\right) \cdot MPa$$

$$P_{ts}(fl, t, s) := \text{rp\_pts}\left(fl, \frac{t}{K}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot MPa$$

$$\rho_{th}(fl, t, h, r) := \text{rp\_rhoth}\left(fl, \frac{t}{K}, h \cdot \frac{kg}{kJ}, r\right) \cdot \frac{kg}{m^3}$$

$$\rho_{ts}(fl, t, s) := \text{rp\_rhots}\left(fl, \frac{t}{K}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kg}{m^3}$$

$$\rho_{ph}(fl, p, h) := \text{rp\_rhoph}\left(fl, \frac{p}{MPa}, h \cdot \frac{kg}{kJ}\right) \cdot \frac{kg}{m^3}$$

$$\rho_{ps}(fl, p, s) := \text{rp\_rhops}\left(fl, \frac{p}{MPa}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kg}{m^3}$$

$$h_{ps}(fl, p, s) := \text{rp\_hps}\left(fl, \frac{p}{MPa}, s \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kJ}{kg}$$

$$s_{ph}(fl, p, h) := \text{rp\_sph}\left(fl, \frac{p}{MPa}, h \cdot \frac{kg}{kJ}\right) \cdot \frac{kJ}{kg \cdot K}$$

$$h_{ts}(fl, t, h) := \text{rp\_hts}\left(fl, \frac{t}{K}, h \cdot \frac{kg \cdot K}{kJ}\right) \cdot \frac{kJ}{kg}$$

$$s_{th}(fl, t, h, r) := \text{rp\_sth}\left(fl, \frac{t}{K}, h \cdot \frac{kg}{kJ}, r\right) \cdot \frac{kJ}{kg \cdot K}$$

Some Additional Unit Definitions

$kJ \equiv 1000 \cdot J$ $\%\% \equiv 1\%$

$\mu Pa \equiv 10^{-6} \cdot Pa$

Some Additional Utility Functions

Utility function that inputs a two-column matrix of fluids and mole fractions and outputs a formatted custom fluid string for the mixtures.

```
MixString(M) :=
    n ← rows(M) − 1
    s ← ""
    y ← 0
    for i ∈ 0 .. n
        y ← y + M_{i,1}
        x ← concat("[", num2str(M_{i,1}), "]")
        s ← concat(s, M_{i,0}, x)
        if i < n
            s ← concat(s, "&")
    msg ← "Composition does not sum to unity."
    if |1 − y| < 0.000001
        s
    else
        error(msg)
```

Appendix D - RefProp Speed Up

Minimizing NIST Calls

Each RefProp call passes the name of the desired fluid to the underlying NIST DLL code. The NIST DLL then opens the appropriate fluid property file and extracts the needed physical constants and coefficients to properly calculate the physical properties for the requested fluid.

The NIST code is written in such a way, however, that it checks to see if the requested fluid file is already open. If it is, then no file I/O is necessary and the property calculations proceed directly. That is to say, 20 property calls for the same fluid, say "CO2.fld", only results in a single property file I/O request during the first property call.

Users can enhance the efficiency of the RefProp Add-In by avoiding repetitive calls to alternating fluid types. For example, if properties for CO2 and Water are desired within a programming loop, it would be more efficient to make all of the property calls for Water, followed by all the property calls for "CO2.fld"; resulting in just two property file I/O calls per programming loop.

Alternatively, making twenty property calls in matched pairs (one call for each fluid) would result in 20 times that many fluid property file I/O requests (40) per programming loop and would be ill advised.

Local Libraries

Since version 2.0 of the RefProp Add-in wrapper, the software looks on the user's hard drive for the NIST REFPROP installation directory. The fluid and mixture files are located underneath that installation directory and this is the best location for fastest retrieval. Should there be a need to access custom fluid files on another path, say a departmental network share drive, it is best if that share drive server is local to the site where calculations are being performed.