Prode Properties

Properties of pure fluids and mixtures

User’s Manual rel. 1.28

www.prode.com

License agreement

Agreement made between Prode "Prode" and "User".

• Prode is the owner of the product "Prode Properties" including , but not limited to, dynamic link libraries, static libraries, header files, sample programs, utility programs, together with the accompanying documentation collectively known as the "software",

• User desires to obtain the right to utilize the software, the parties hereby agree as follows

Personal license

A version with limited features is available for personal use at home or in educational establishments for teaching purposes, all other applications, without first obtaining a commercial license from Prode, are expressly prohibited.

Commercial license

Upon full payment of the license fee the User has full right to utilize the purchased number of units of the software, a unit is defined as one copy of the software or any portion thereof installed on one stand-alone computer, for networked computers one unit shall be applied for each user having concurrent access and one unit shall be applied for the server.

For all applications

• Prode grants the nonexclusive, nontransferable right to use the software.

• User has a royalty free right to reproduce and distribute the software as available from Prode Internet server provided that User doesn't remove or alter any part of the software or of the licensing codes and threat the software as a whole unit.

• You cannot decompile, disassemble or reverse engineer the files containing the licensed software, or any backup copy, in whole or in part.

• You cannot rent, lease or sublicense the Licensed Software without express agreement by Prode.

• The software is provided "as is, where is" , Prode does not warrant that software is free from defects, or that any technical or support services provided by Prode will correct any defects which might exist.

• Prode shall not be liable for any damages that may result directly or indirectly from the use of these software programs including any loss of profits, loss of revenues, loss of data, or any incidental or consequential damages that may arise out of use of these software.

• Your license is effective upon your acceptance of this agreement and installing the Licensed Software.

• This license agreement shall remain in effect until the Licensed Software will be in use.

• You may terminate it at any time by destroying the Licensed Software together with all copies. It will also terminate if you fail to comply with any term or condition of this Agreement. You agree upon such termination to destroy all copies of the Licensed Software in any form in your possession or under your control.

• Prode will provide the licensee with limited technical support by telephone, or by electronic media for a period of 60 days after delivery of the product.

How to contact Prode

you can contact Prode by phone, web page or email, the details are available at http://www.prode.com

How to obtain technical support

we welcome your comments or suggestions about our products , while the program has been tested carefully to ensure proper operation, it still may be possible for an unusual situation to result in an error. We will have a much greater chance of fixing or assisting with errors and problems if they are provided to us in a form that is repeatable. In reporting a problem to us, the following information should be given:

• customer reference

• the version of the software

• a copy of the procedure you are running and if possible the input data

• a detailed description of what you were doing (sequence of operations) when the problem occurred

• any additional information you think may describe the problem

Introduction to Prode Properties

Prode Properties includes a comprehensive collection of procedures to solve problems in areas such as :

- Process Simulation, Process Control,
- Physical Properties Data, Data Analysis
- Equipment Design, Separations…

Technical features overview

- Prode Properties is a thermodynamic library written in C++ (ISO 2017) and released as compiled library
- Multiple threads are supported by design, no limits on number of concurrent threads
- Standard versions for Window, Linux and Android, several other versions including WebAssembly available on request
- Allows up to 100 streams with up to 100 components per stream (user can redefine)
- Several compilations of chemical data and BIPs available (user can edit / add new components and BIPs)
- Comprehensive set of thermodynamic models
- Complete set of flash operations T-P, H-P, H-T, S-P, S-T, V-P, V-T, H-V, S-V, H-S, constant energy, phase-fraction...
- Functions for calculating specific properties of mixtures (critical point, Cricodentherm, Cricondenbar, cloud point etc.)
- Functions for calculating fugacity, enthalpy, entropy, volume plus derivatives vs. temperature, pressure, composition
- Functions for calculating equilibrium lines at specified phase fractions (generation of phase diagrams)
- Functions for solving operating blocks as mixer, gas separator, liquid separator, distillation column, compressor, piping
- Functions for calculating stream properties as density, conductivity, viscosity (gaseous and liquid phases) surface tension, speed of sound, Joule Thomson etc.

Reference Literature

Although Prode Properties may appear easy to utilize, a basic knowledge of applied thermodynamics is required for selecting the proper methods and critically evaluate the results, to support your work we suggest these books :

- Introduction to Chemical Engineering Thermodynamics by Smith, Van Ness, Abbott
- The Properties of Gases & Liquids, by Reid, Prausnitz, Poling
- Thermodynamic models for industrial applications by Kontogeorgis, Folas

History

version 1.01 (first commercial release of Prode Properties) distributed in 1992

Actual release

version 1.28 , released on Dec. 2022

Roadmap

version 1.29 (new version including additional models and features) 2024

Install the software

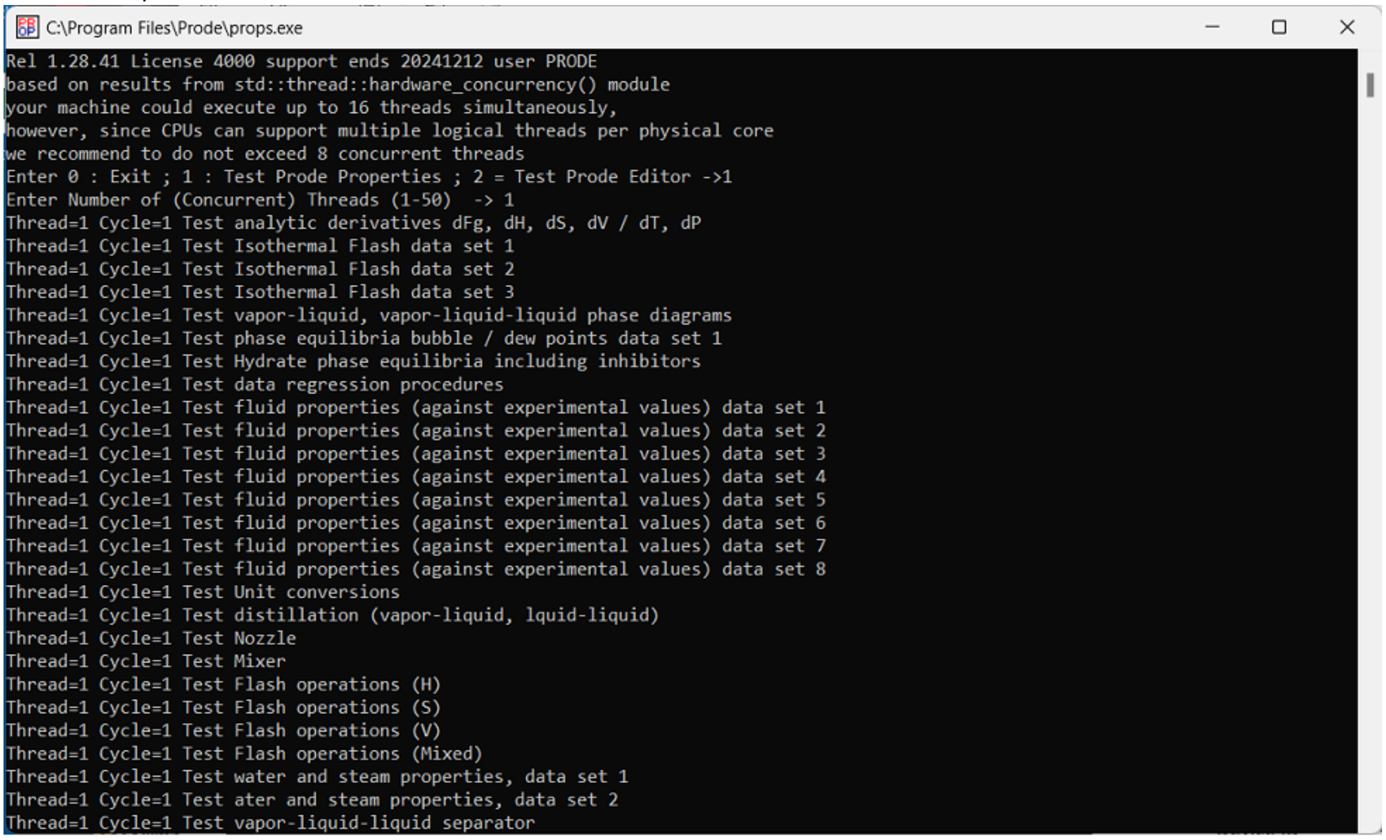
Prode Properties is available from this page

http://www.prode.com/en/download.htm

download the software, run the installer and follow the instruction
ask Prode for a different versions (there are versions for Windows, Linux, Android, IOS)

Test the software

Prode Properties includes a desktop app (Prode Properties Tests) , run the app and follow the instruction
Enter 1 to start a series of tests (the user can select single or multiple threads and compare results)
Enter 2 to open Prode Editor



```
Rel 1.28.41 License 4000 support ends 20241212 user PRODE
based on results from std::thread::hardware_concurrency() module
your machine could execute up to 16 threads simultaneously,
however, since CPUs can support multiple logical threads per physical core
we recommend to do not exceed 8 concurrent threads
Enter 0 : Exit ; 1 : Test Prode Properties ; 2 = Test Prode Editor ->1
Enter Number of (Concurrent) Threads (1-50)  -> 1
Thread=1 Cycle=1 Test analytic derivatives dFg, dH, dS, dV / dT, dP
Thread=1 Cycle=1 Test Isothermal Flash data set 1
Thread=1 Cycle=1 Test Isothermal Flash data set 2
Thread=1 Cycle=1 Test Isothermal Flash data set 3
Thread=1 Cycle=1 Test vapor-liquid, vapor-liquid-liquid phase diagrams
Thread=1 Cycle=1 Test phase equilibria bubble / dew points data set 1
Thread=1 Cycle=1 Test Hydrate phase equilibria including inhibitors
Thread=1 Cycle=1 Test data regression procedures
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 1
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 2
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 3
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 4
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 5
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 6
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 7
Thread=1 Cycle=1 Test fluid properties (against experimental values) data set 8
Thread=1 Cycle=1 Test Unit conversions
Thread=1 Cycle=1 Test distillation (vapor-liquid, lquid-liquid)
Thread=1 Cycle=1 Test Nozzle
Thread=1 Cycle=1 Test Mixer
Thread=1 Cycle=1 Test Flash operations (H)
Thread=1 Cycle=1 Test Flash operations (S)
Thread=1 Cycle=1 Test Flash operations (V)
Thread=1 Cycle=1 Test Flash operations (Mixed)
Thread=1 Cycle=1 Test water and steam properties, data set 1
Thread=1 Cycle=1 Test ater and steam properties, data set 2
Thread=1 Cycle=1 Test vapor-liquid-liquid separator
```

With option 1) the procedure executes a series of automatic tests (solving hundreds of predefined problems) then it reports errors and problems.
The user can start multiple threads and compare results, note that Prode Properties supports any number of concurrent threads,
the table shows the number of seconds required to complete the tests on a Windows 11-64 computer with CPU AMD 4750U

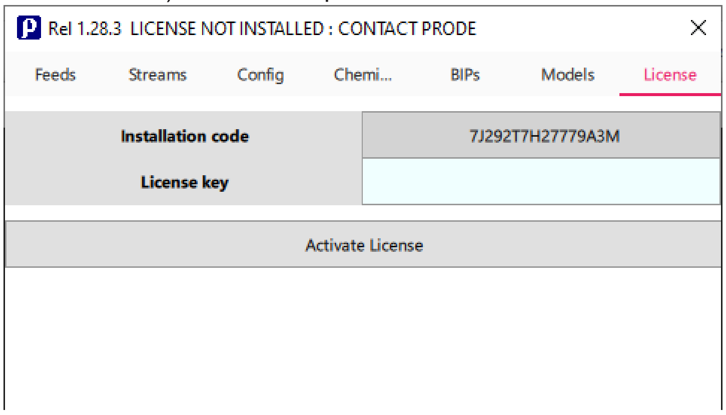
Nr. threads	Time (seconds)	relative speed (Time single thread * number of threads) / Time multi-threaded cycle
1	5	1
2	5	2
4	7	2.85
8	10	4
16	25	3.2
32	49	3.2

the CPU AMD 4750U has 8 physical cores, the results show that relative speed do not increase for any number of threads > 8

Running the same tests on a CPU with 16 physical cores the application returns a relative speed of about 8 with 16 threads, which means that you can execute up tp 8 times faster than the single thread approach.

Require a software license

• In Prode Properties Tests application enter 2 to open Prode Editor
• the Editor will show the License tab, copy the ID or installation code (to copy a value select the cell and use the right button on the mouse) in this example the ID is 7J292T7H27779A3M

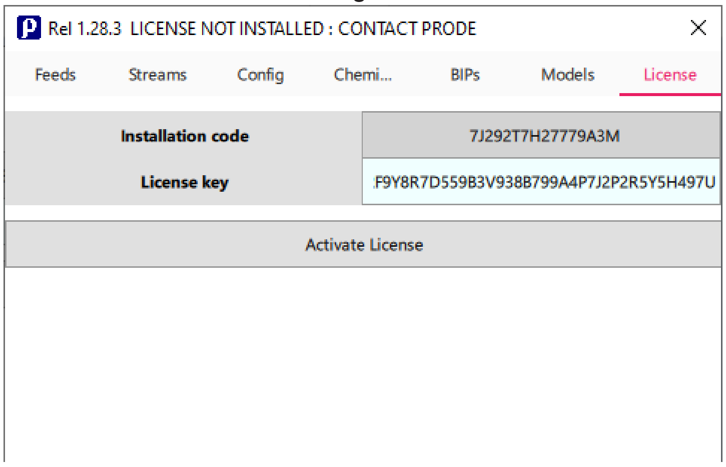

Rel 1.28.3 LICENSE NOT INSTALLED : CONTACT PRODE

Installation code	7J292T7H27779A3M
License key	

• Contact Prode to receive a software license : email us the installation code and the application (commercial use or nonprofit educational institution)

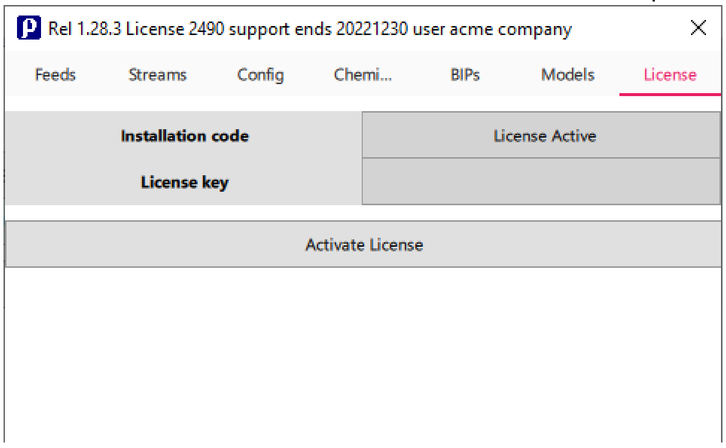
Activate a software license

After the order you will receive a license key
• Open Prode Editor, select the Config tab and enter the License Key, you can also copy / paste the value : select the cell with the mouse and use the right button.


Rel 1.28.3 LICENSE NOT INSTALLED : CONTACT PRODE

Installation code	7J292T7H27779A3M
License key	F9Y8R7D559B3V938B799A4P7J2P2R5Y5H497U

• Press the button Activate License, the software will report “License Active”

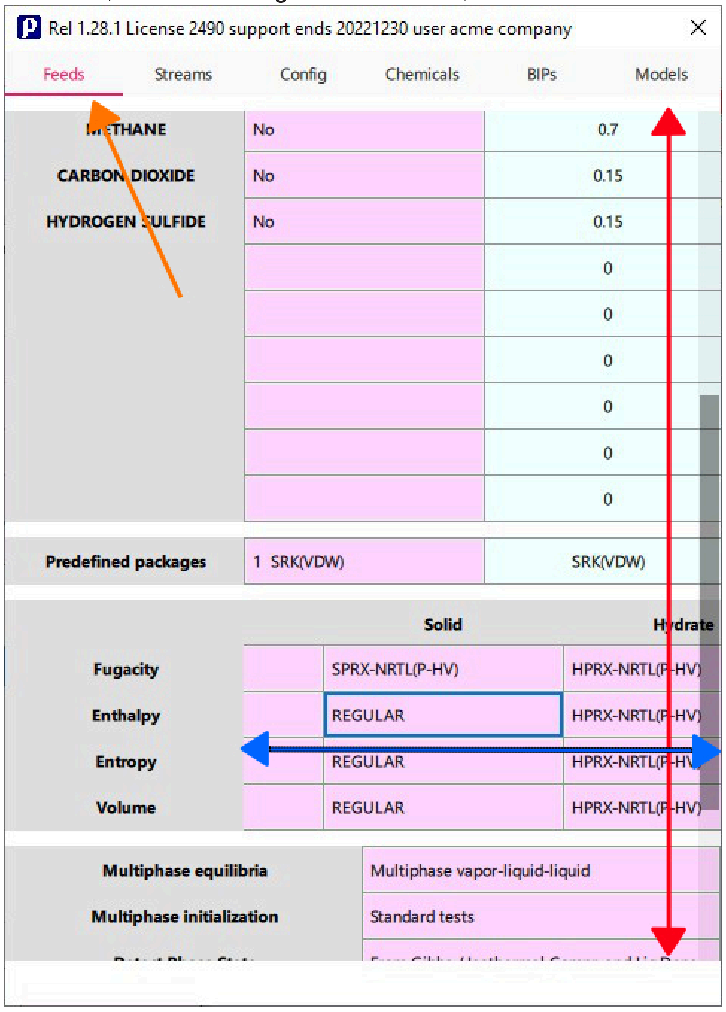

Rel 1.28.3 License 2490 support ends 20221230 user acme company

Installation code	License Active
License key	

Prode Editor : Quick Start guide

Prode Properties includes a editor with 6 pages

- Feeds, to edit / change compositions, flows, models, BIPs
- Streams, to inspect streams, set operating conditions, solve unit operations
- Configuration, to define the units of measurement and settings
- Chemicals, to edit / change chemical's data, use data regression utility to calculate new values, add new chemicals
- BIPs, to edit / change bip's data, use data regression utility to calculate new BIPs
- Models, to edit / change model's data, add new models and chemicals



Rel 1.28.1 License 2490 support ends 20221230 user acme company

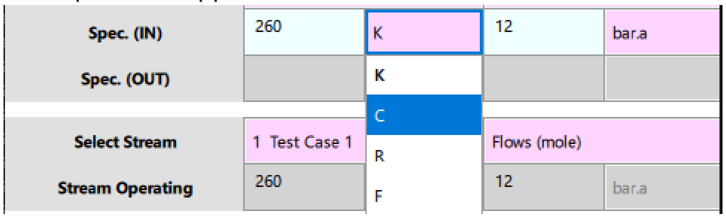
METHANE	No	0.7
CARBON DIOXIDE	No	0.15
HYDROGEN SULFIDE	No	0.15
		0
		0
		0
		0
		0
		0
Predefined packages	1 SRK(VDW)	SRK(VDW)

		Solid	Hydrate
Fugacity		SPRX-NRTL(P-HV)	HPRX-NRTL(P-HV)
Enthalpy		REGULAR	HPRX-NRTL(P-HV)
Entropy		REGULAR	HPRX-NRTL(P-HV)
Volume		REGULAR	HPRX-NRTL(P-HV)

Multiphase equilibria	Multiphase vapor-liquid-liquid
Multiphase initialization	Standard tests

Prode Editor adopts a portable (Windows, Linux, Android, IOS) GUI based on a tabbed dialog with flickable grids.
The elements can be dragged and flicked causing the views to scroll, you can drag the view by pressing and holding a mouse button while moving the cursor, in addition there is a standard scroll bar for vertical scroll.

Unit conversion and data validation
Grids provide support for data validation and unit conversion, to convert to different units select a value in drop-down list



Spec. (IN)	260	K	12	bar.a
Spec. (OUT)		K		
Select Stream	1 Test Case 1	C	Flows (mole)	
Stream Operating	260	R	12	bar.a
		F		

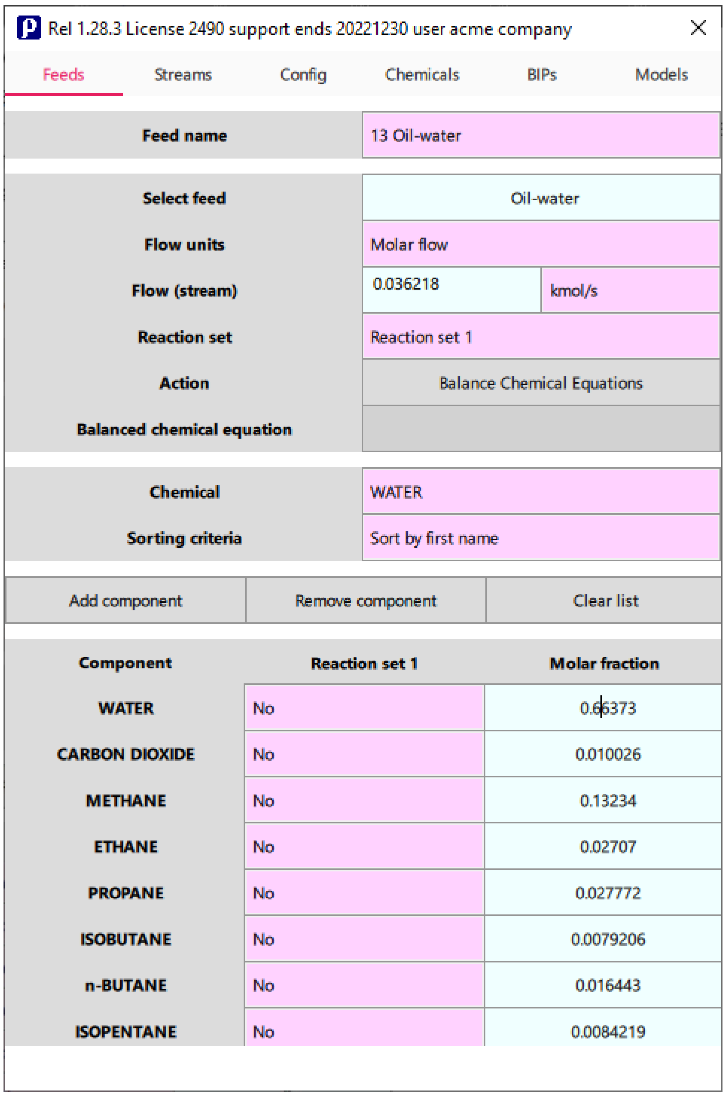
Copy / paste operations
Copy / paste operations are allowed, use the mouse right button over a cell to activate this option (available for cells containing data inputs or results)

Change sorting criteria in combo selectors
Combo selectors (for the lists of chemicals) have two indexing options (sort by name or formula) and a quick access mechanism, typing a letter the list will scroll to the first matching value.

Reports for warnings and errors
Messages with warnings and errors are visible at the bottom of dialog, click on to delete

Prode Editor : Feeds page

From this page you can :
• select a stream (select / edit stream)
• edit / change name, stream’s flow, the list of components and relative weights
• define reaction sets (for reactive flash operations), balance chemical equations for the different reaction sets
• define the models for different properties (Fg,H,S,V…), define the different settings, select BIPs dataset (VLE, LLE…)
• save / store the edited stream

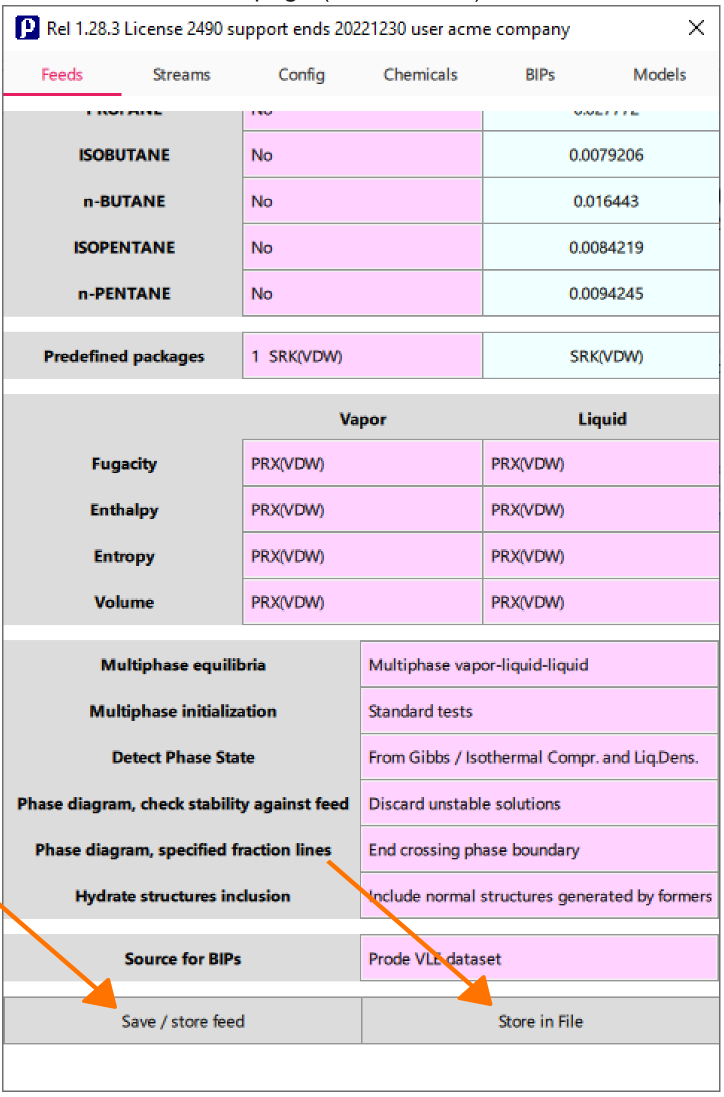

Rel 1.28.3 License 2490 support ends 20221230 user acme company
Feeds
Streams
Config
Chemicals
BIPs
Models
Balance Chemical Equations
Add component
Remove component
Clear list

Feed name	13 Oil-water
Select feed	Oil-water
Flow units	Molar flow
Flow (stream)	0.036218 kmol/s
Reaction set	Reaction set 1
Action	Balance Chemical Equations
Balanced chemical equation	
Chemical	WATER
Sorting criteria	Sort by first name

Component	Reaction set 1	Molar fraction
WATER	No	0.66373
CARBON DIOXIDE	No	0.010026
METHANE	No	0.13234
ETHANE	No	0.02707
PROPANE	No	0.027772
ISOBUTANE	No	0.0079206
n-BUTANE	No	0.016443
ISOPENTANE	No	0.0084219

• use Select feed to select a stream, you can define stream’s name, flow and units (molar or mass)
• select components from the list of chemicals and Add component / Remove component / Clear list to define composition, select Sorting criteria to obtain lists sorted by name or formula, with chemical’s list open type the first character in your chemical to scroll to the first matching value
• For Reactive Flash operations you can define up to 5 reaction sets per stream, each components can be included as reactant, product or neutral (no), on each reaction set select Balance Chemical Equation button to obtain the chemical equation

Prode Editor : Feeds page (continuation)


Rel 1.28.3 License 2490 support ends 20221230 user acme company
Feeds
Streams
Config
Chemicals
BIPs
Models

ISOBUTANE	No	0.0079206
n-BUTANE	No	0.016443
ISOPENTANE	No	0.0084219
n-PENTANE	No	0.0094245
Predefined packages	1 SRK(VDW)	SRK(VDW)

	Vapor	Liquid
Fugacity	PRX(VDW)	PRX(VDW)
Enthalpy	PRX(VDW)	PRX(VDW)
Entropy	PRX(VDW)	PRX(VDW)
Volume	PRX(VDW)	PRX(VDW)

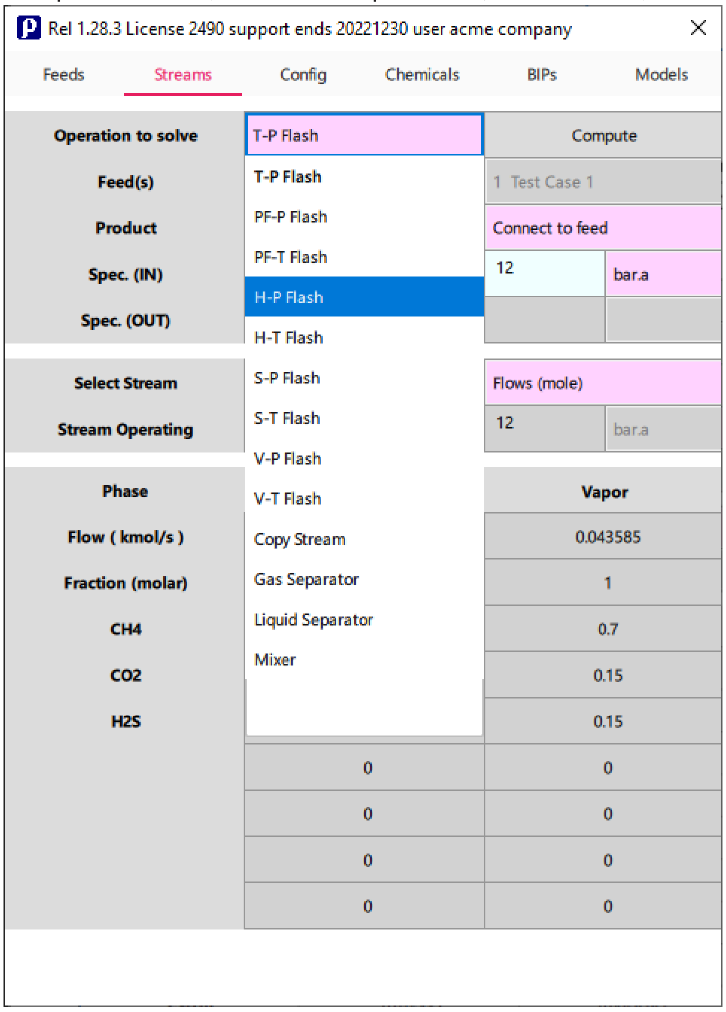
Multiphase equilibria	Multiphase vapor-liquid-liquid
Multiphase initialization	Standard tests
Detect Phase State	From Gibbs / Isothermal Compr. and Liq.Dens.
Phase diagram, check stability against feed	Discard unstable solutions
Phase diagram, specified fraction lines	End crossing phase boundary
Hydrate structures inclusion	Include normal structures generated by formers
Source for BIPs	Prode VLE dataset

• Define the models for the different properties (fugacity, enthalpy, entropy, volume) and state (vapor, liquid, solid, hydrate) you can select from the lists or use one of predefined packages
• the editor allows to edit existing packages and define new packages, select a package in the list, the models you wish to define, enter a name for the package and use button Save to store the new package
• Define settings
• Multiphase equilibria, allows to define different solutions as vapor-liquid, vapor-liquid-liquid and vapor-liquid-solid
• Multiphase initialization, allows to reduce the number of trial phases thus reducing time required
• Detect Phase State, allows to use different methods to detect the state of each phase
• Phase diagram, check stability against feed, allows to include stability analysis on each calculated point
• Phase diagram, specified phase fraction lines, allows to terminate lines when crossing a phase boundary
• Hydrates structures inclusion, allows to test all possible hydrate structures which may be generated by former(s)
• Select the source for BIPs
Prode VLE dataset, includes a large collection of BIPS calculated from VLE data points
Prode LLE dataset, includes a limited number of BIPS calculated from LLE data points
Prode SLE dataset, includes a limited number of BIPS calculated from SLE data points
Prode Hydrate dataset, includes a limited number of BIPS specific for hydrate phase equilibria
• Select Save / store feed button, the program will store your data
• Select Store in file button, the program will save your data in file

Prode Editor : Streams page

From this page you can :

• Inspect streams, solve flash operations, mix streams, solve vapor-liquid, liquid-liquid separations



Rel 1.28.3 License 2490 support ends 20221230 user acme company

Feeds | Streams | Config | Chemicals | BIPs | Models

Operation to solve	T-P Flash	Compute	
Feed(s)	T-P Flash	1 Test Case 1	
Product	PF-P Flash	Connect to feed	
Spec. (IN)	PF-T Flash	12	bar.a
Spec. (OUT)	H-P Flash		
	H-T Flash		
Select Stream	S-P Flash	Flows (mole)	
Stream Operating	S-T Flash	12	bar.a
	V-P Flash		
Phase	V-T Flash	Vapor	
Flow (kmol/s)	Copy Stream	0.043585	
Fraction (molar)	Gas Separator	1	
CH4	Liquid Separator	0.7	
CO2	Mixer	0.15	
H2S		0.15	
	0	0	
	0	0	
	0	0	
	0	0	

Inspect a stream

• use Select Stream to select a stream, note that depending from selected option (connect to feed, connect to product or do no connect) the selection may change when the feed or product change

Compute flash operations

• make sure all feeding streams have been defined

• select feeding streams, product stream and the operation to solve, there is an option to connect the selected stream (and product) to feed or product (to view results)

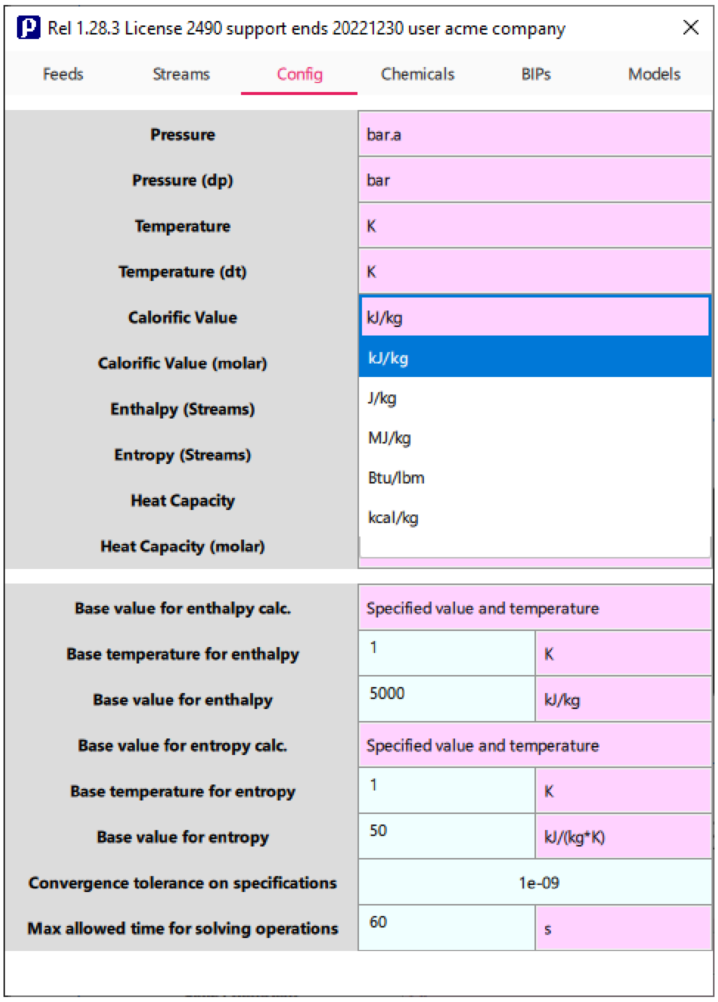
• enter the required specifications and select “Compute”

List of operations which you can solve from Prode Editor
Flash at specified Temperature and Pressure
Flash at specified Liquid Fraction and Pressure or Temperature
Flash at specified Enthalpy and Pressure or Temperature
Flash at specified Entropy and Pressure or Temperature
Flash at specified Volume and Pressure or Temperature
Copy Streams
Vapor-Liquid and Liquid-Liquid separators
Mixers

**Customized versions can include additional operations

Prode Editor : Config page

From this page you can define
• units of measurement
• parameters, options and preferences (settings) utilized by Prode Properties


Rel 1.28.3 License 2490 support ends 20221230 user acme company
Feeds
Streams
Config
Chemicals
BIPs
Models

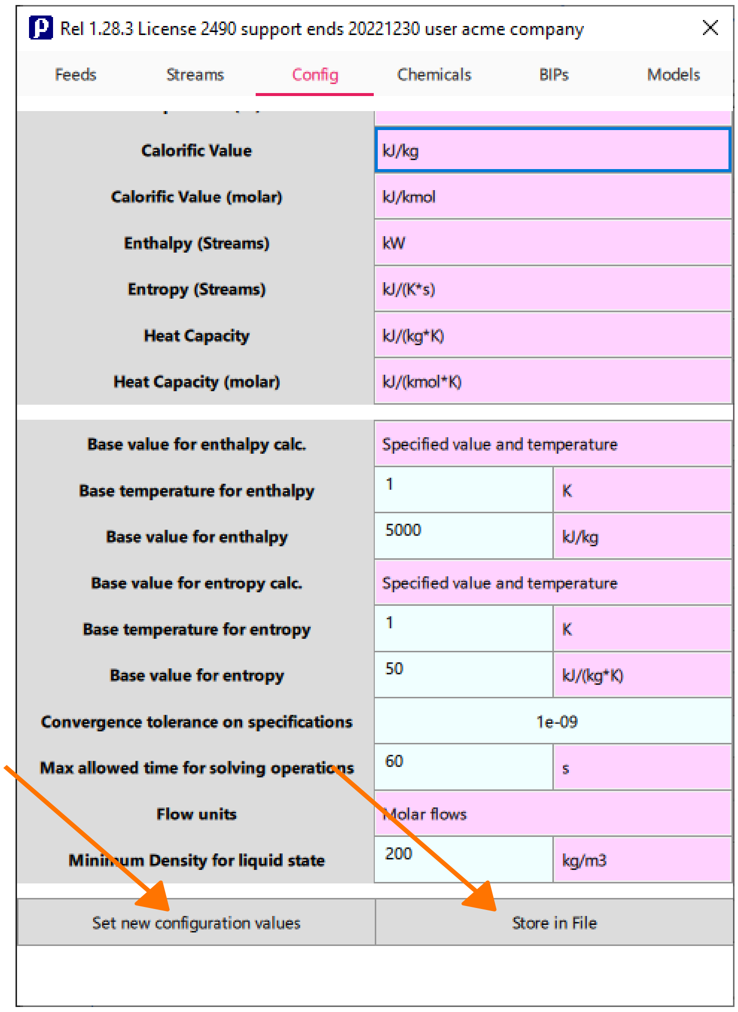
Pressure	bar.a	
Pressure (dp)	bar	
Temperature	K	
Temperature (dt)	K	
Calorific Value	kJ/kg	
Calorific Value (molar)		
Enthalpy (Streams)		
Entropy (Streams)		
Heat Capacity		
Heat Capacity (molar)		
Base value for enthalpy calc.	Specified value and temperature	
Base temperature for enthalpy	1	K
Base value for enthalpy	5000	kJ/kg
Base value for entropy calc.	Specified value and temperature	
Base temperature for entropy	1	K
Base value for entropy	50	kJ/(kg*K)
Convergence tolerance on specifications	1e-09	
Max allowed time for solving operations	60	s

kJ/kg
J/kg
MJ/kg
Btu/lbm
kcal/kg

Setting the units of measurement

With Prode Properties you have complete control over the engineering units
• select your preferred units from the list available for each property
• select Set new configuration values button to update configuration, the program will convert automatically the input values and the results accordingly

Prode Editor : Config page (continuation)


Rel 1.28.3 License 2490 support ends 20221230 user acme company
Feeds
Streams
Config
Chemicals
BIPs
Models
Set new configuration values
Store in File

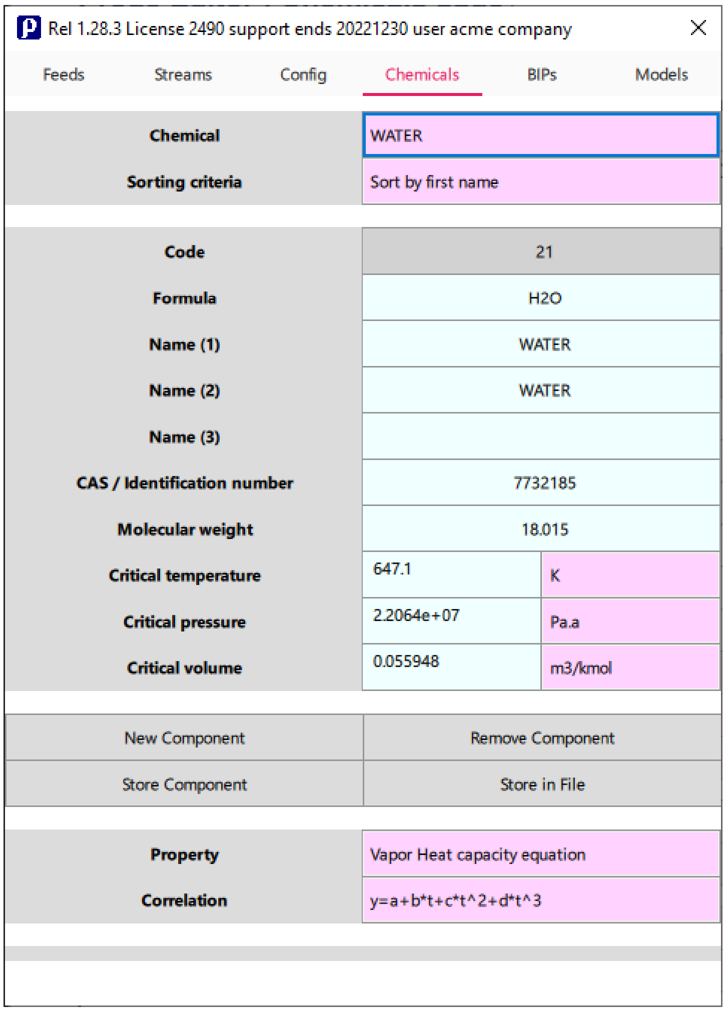
Calorific Value	kJ/kg	
Calorific Value (molar)	kJ/kmol	
Enthalpy (Streams)	kW	
Entropy (Streams)	kJ/(K*s)	
Heat Capacity	kJ/(kg*K)	
Heat Capacity (molar)	kJ/(kmol*K)	
Base value for enthalpy calc.	Specified value and temperature	
Base temperature for enthalpy	1	K
Base value for enthalpy	5000	kJ/kg
Base value for entropy calc.	Specified value and temperature	
Base temperature for entropy	1	K
Base value for entropy	50	kJ/(kg*K)
Convergence tolerance on specifications	1e-09	
Max allowed time for solving operations	60	s
Flow units	Molar flows	
Minimum Density for liquid state	200	kg/m3

configurable parameters :

• max number of streams
• max number of components per stream
• max number of interaction coefficients pairs per stream
• reference temperature and pressure
• base values for enthalpy and entropy calc's
• convergence tolerance
• max allowed time for solving a operation
• Flow units
• minimum liquid density to validate liquid phase
• select Set new configuration values button to update configuration, the program will adopt the new configuration parameters
• select Store in file button to store actual configuration in file, the program will adopt the new configuration parameters as default values

Prode Editor : Chemicals page

From this page you can :
• Inspect / edit physical properties data stored in the software, regress raw data, add / remove components


Rel 1.28.3 License 2490 support ends 20221230 user acme company

Chemical	WATER
Sorting criteria	Sort by first name

Code	21	
Formula	H2O	
Name (1)	WATER	
Name (2)	WATER	
Name (3)		
CAS / Identification number	7732185	
Molecular weight	18.015	
Critical temperature	647.1	K
Critical pressure	2.2064e+07	Pa.a
Critical volume	0.055948	m3/kmol

Property	Vapor Heat capacity equation
Correlation	y=a+b*t+c*t^2+d*t^3

Inspect / edit data :

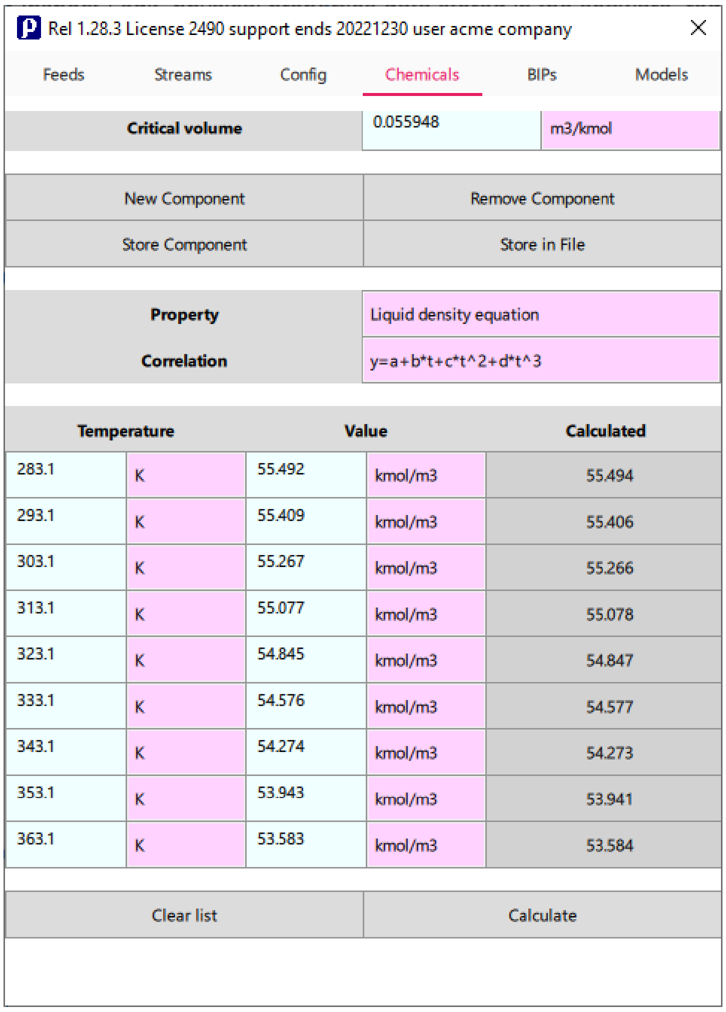
• select the component from the component's list
• edit / modify the related fields
• select "Store Component" button to save the modified data

Adding a new component :
• select "New Component" button
• edit the related fields
• select "Store Component" button to save data

Remove a component :
• select a component from the component's list
• select "Remove Component" button

Update the files which stores physical properties data :
• select the "Save File" button, this command overwrites the file chem.dat , if required you can create a backup

Prode Editor : Chemicals page (continuation)



Rel 1.28.3 License 2490 support ends 20221230 user acme company

Critical volume	0.055948	m3/kmol

Property	Liquid density equation
Correlation	y=a+b*t+c*t^2+d*t^3

Temperature		Value		Calculated
283.1	K	55.492	kmol/m3	55.494
293.1	K	55.409	kmol/m3	55.406
303.1	K	55.267	kmol/m3	55.266
313.1	K	55.077	kmol/m3	55.078
323.1	K	54.845	kmol/m3	54.847
333.1	K	54.576	kmol/m3	54.577
343.1	K	54.274	kmol/m3	54.273
353.1	K	53.943	kmol/m3	53.941
363.1	K	53.583	kmol/m3	53.584

Note : Prode Properties supports more than 15 different correlations per each property, you can select the correlation which best fits experimental data

Regress raw data

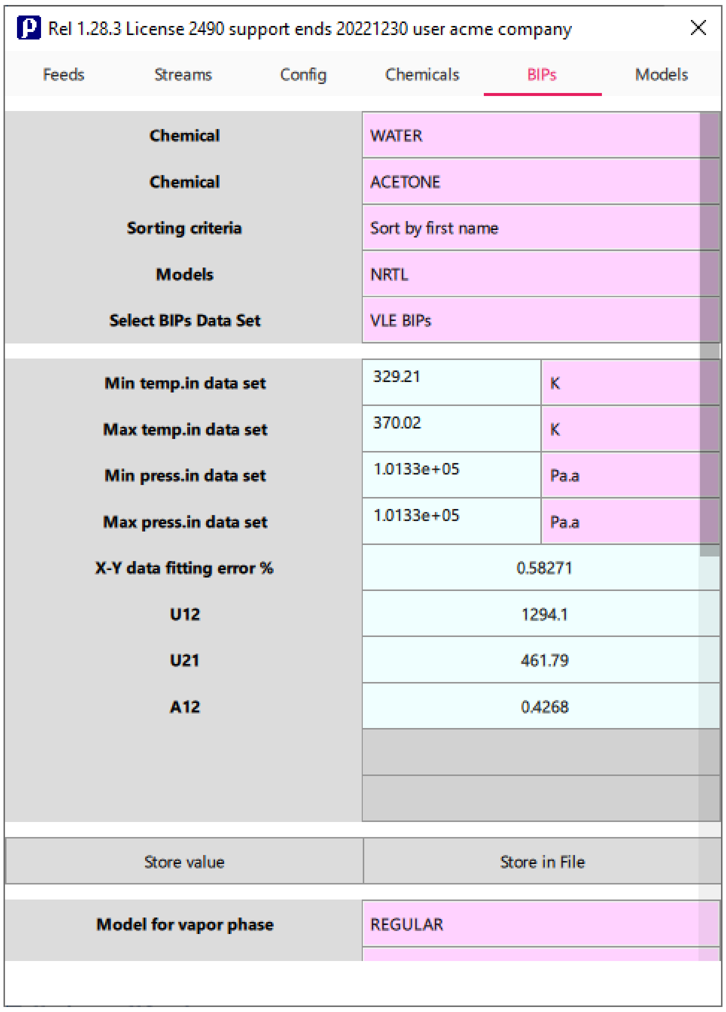
- select a chemical
- select a property and the correlation for fitting raw data
- enter the available data (all temperature and value pairs) with the proper units of measurement
- select Calculate button , the procedure adds the calculated parameters to the database
- evaluate calculated values and errors, you may try different correlations for best data fitting
- select “Store Component” button to save the new data

Update the file which stores physical properties data :

- select “Save File” button, this command overwrites the file chem.dat , if required you can create a backup

Prode Editor : BIPs page

From this page you can :
• edit Binary Interaction Parameters
• add / remove Binary Interaction Parameters
• regress VLE (vapor-liquid) , LLE (liquid-liquid) , SLE (solid-liquid) data points
• save all data in a file

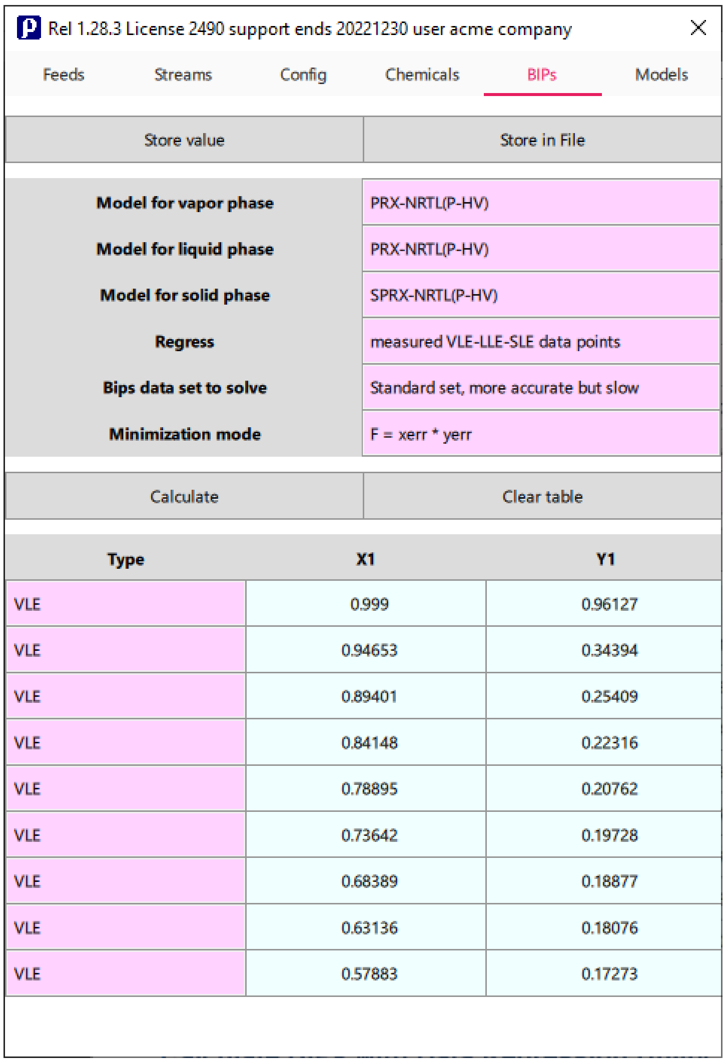

Rel 1.28.3 License 2490 support ends 20221230 user acme company
Feeds
Streams
Config
Chemicals
BIPs
Models
Store value
Store in File

Chemical	WATER	
Chemical	ACETONE	
Sorting criteria	Sort by first name	
Models	NRTL	
Select BIPs Data Set	VLE BIPs	
Min temp.in data set	329.21	K
Max temp.in data set	370.02	K
Min press.in data set	1.0133e+05	Pa.a
Max press.in data set	1.0133e+05	Pa.a
X-Y data fitting error %	0.58271	
U12	1294.1	
U21	461.79	
A12	0.4268	
Model for vapor phase	REGULAR	

Edit / modify data :

• select two components from the component’s lists
• select the database (VLE/LLE/SLE/Hydrate)
• select the model
• edit / modify BIPs
• select “Store value” button to save the modified data
Update the file which stores physical properties data :
• select “Store in File” button, this command overwrites the file bips.dat

Prode Editor : BIPs page (continuation)



Rel 1.28.3 License 2490 support ends 20221230 user acme company

Feeds | Streams | Config | Chemicals | BIPs | Models

Store value | Store in File

Model for vapor phase	PRX-NRTL(P-HV)
Model for liquid phase	PRX-NRTL(P-HV)
Model for solid phase	SPRX-NRTL(P-HV)
Regress	measured VLE-LLE-SLE data points
Bips data set to solve	Standard set, more accurate but slow
Minimization mode	F = xerr * yerr

Calculate | Clear table

Type	X1	Y1
VLE	0.999	0.96127
VLE	0.94653	0.34394
VLE	0.89401	0.25409
VLE	0.84148	0.22316
VLE	0.78895	0.20762
VLE	0.73642	0.19728
VLE	0.68389	0.18877
VLE	0.63136	0.18076
VLE	0.57883	0.17273

Calculate BIPs with Data Regression Utility

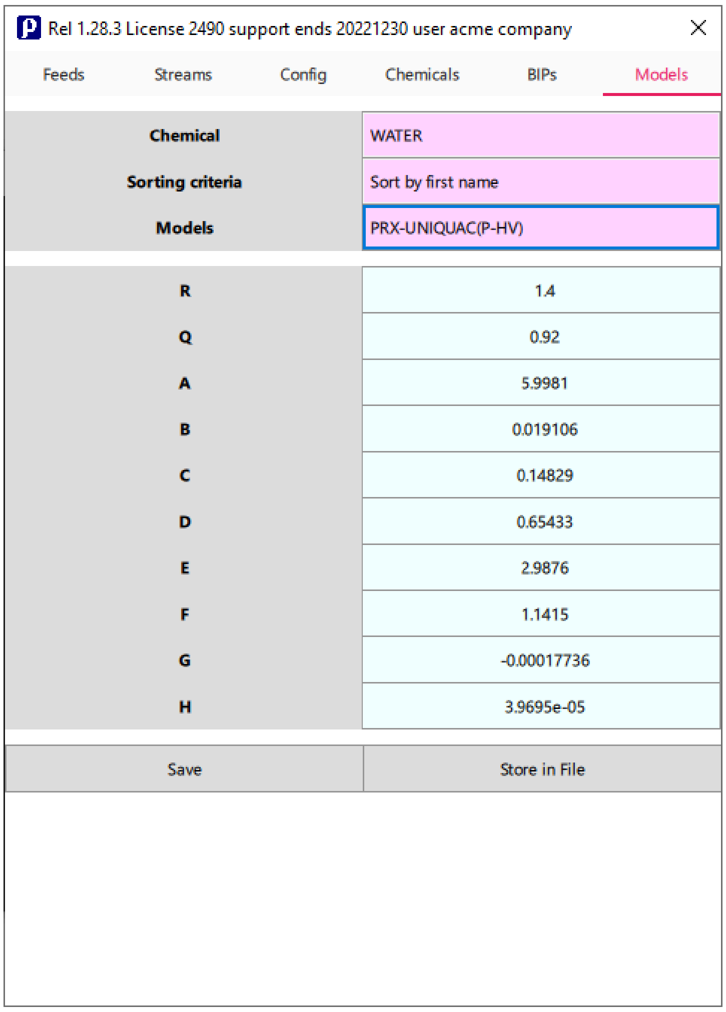
Enter experimental VLE-LLE-SLE data points or generate VLE points with a predictive model
• select the Chemicals
• select BIPs Data Set
• select the Models for vapor, liquid, solid phases
• select the type of data points
• select the BIPs data set to solve
• select Minimization mode

• If you have selected Regress measured VLE-LLE-SLE data points enter one point per row,
    in Type select VLE / LLE / SLE
    in X1, Y1 enter the molar fractions of first component C1 in the different phases,
        for VLE : liquid in X1, vapor in Y1,
        for LLE : liquid (phase 1) in X1, liquid (phase 2) in Y1,
        for SLE : liquid in X1, solid in Y1,
    enter the temperature and the pressure for that point.
• If you have selected Regress VLE points calculated with UNIFAC the procedure will calculate the required VLE points

• select Calculate button , the procedure adds the calculated BIP values to the database
• select “Store Value” button to save the new data
• select “Save File” button to save these values in bips.dat.file

Prode Editor : Models page

From this page you can :
• edit the parameters required by the different models available in library



Rel 1.28.3 License 2490 support ends 20221230 user acme company

Chemical	WATER
Sorting criteria	Sort by first name
Models	PRX-UNIQUAC(P-HV)

R	1.4
Q	0.92
A	5.9981
B	0.019106
C	0.14829
D	0.65433
E	2.9876
F	1.1415
G	-0.00017736
H	3.9695e-05

Edit / modify data :

• select the component from the component’s lists
• select the model
• edit / modify the parameters
• select the “Save” button to save the modified data
• select “Save File” button to save these values in mod.dat.file

Prode Properties : initial setup

This section provides important information about Prode Properties initial settings.

Locating the files

The installation procedure creates different folders for program files and data files

Program files folder (Windows version)
C:\Program Files\Prode\

Sample files folders (Windows version)

\Prode\C	includes support files for C / C++ applications
\Prode\Excel	includes support files for Microsoft Excel
\Prode\LIB	includes Prode Properties library files
\Prode\LibreOffice	includes support files for LibreOffice applications
\Prode\NET	includes support files for NET applications
\Prode\Python	includes support files for Python applications

Data files folder (Windows version)
C:\ProgramData\prode includes these files
chema.dat
chemb.dat
pseudo.dat
bips.dat
mod.dat
def.ppp
res.lan
cfg.dat

do not remove or rename these files, when the software cannot access these files (for example because they have been disseminated in different directories) an error message "Corrupted file, error reading data file" will be generated.

Make sure all users can access data files folder

When installing Prode Properties for users without full administrative rights make sure all users have read/write rights to data files folder, if a user has no read / write rights on data files folder the program can generate errors and stop working.

Avoid errors in read / write operations

If a user doesn't receive full read / write permissions on data files folder the program can generate a error when saving def.ppp or chem.dat files,

if you see this error you can

1) login as admin, and run Prode Properties

2) immediately before to save def.ppp or chem.dat (from Prode Properties) , with Windows File Manager manually delete the file which you wish to overwrite (def.ppp or chem.dat)

3) (from Prode Properties) save the file

Getting Started from Microsoft Excel (Windows version)

Prerequisites

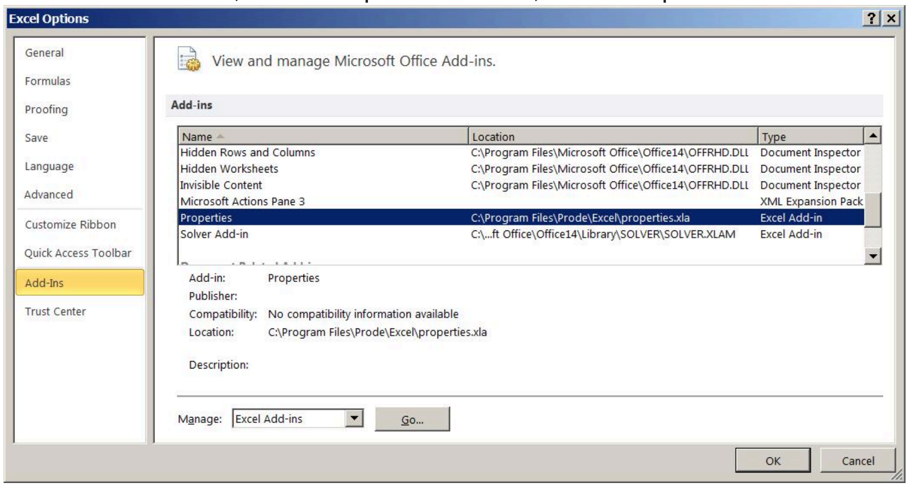
• install the 64 bit version of Excel : the different versions (32 or 64 bit) of Excel require different versions of Prode Properties library, before to install Prode Properties make sure which version of Excel you have.

• verify Windows settings : in Windows open the regional settings dialog to see which separators Excel requires in macros, by default Excel adopts commas and, in macros accessing Prode library, the different parameters must be separated by commas, for example = EStrGD(1,300,1.0E5) returns the gas density for stream 1 at 300 K, 1.0E5 Pa

Install Prode Properties add-in

• before to use Excel you must load the add-in (file properties.xla) which instructs Excel about the methods included in Prode Properties library, you need to go through this procedure only once,

To install the add-in, in Excel open File menu, choose Options item and then Add-Ins

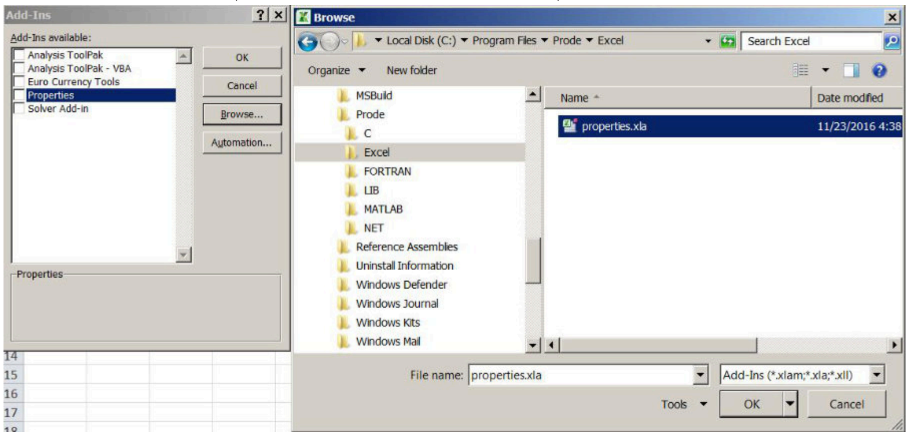


Name	Location	Type
Hidden Rows and Columns	C:\Program Files\Microsoft Office\Office14\OFFRHD.DLL	Document Inspector
Hidden Worksheets	C:\Program Files\Microsoft Office\Office14\OFFRHD.DLL	Document Inspector
Invisible Content	C:\Program Files\Microsoft Office\Office14\OFFRHD.DLL	Document Inspector
Microsoft Actions Pane 3		XML Expansion Pack
Properties	C:\Program Files\Prode\Excel\properties.xla	Excel Add-in
Solver Add-in	C:\...ft Office\Office14\Library\SOLVER\SOLVER.XLAM	Excel Add-in

Add-in: Properties
Publisher:
Compatibility: No compatibility information available
Location: C:\Program Files\Prode\Excel\properties.xla

Description:

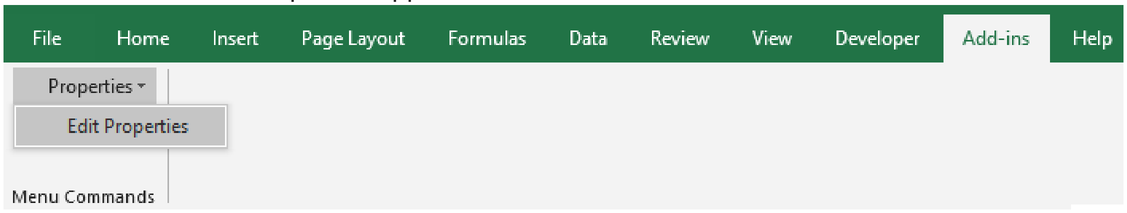
• on the bottom select Manage Excel Add-Ins and click Go, you'll see a list of add-ins, some checked, some not checked. If Prode Properties isn't listed (and it won't be unless you went through this procedure earlier) browse for the properties.xla file (by default installed in C:\Program Files\Prode\Excel\) then back your way out. Now Prode Properties should be listed in the list of add-ins, its box should be checked, click Ok to exit Excel Add-ins dialog,



File name: properties.xla
Add-Ins (*.xlam;*.xla;*.xll)

• Close dialogs with Ok button, this completes the procedure to install the add-in

The menu for Prode Properties appears under Add-Ins tab in Microsoft Excel



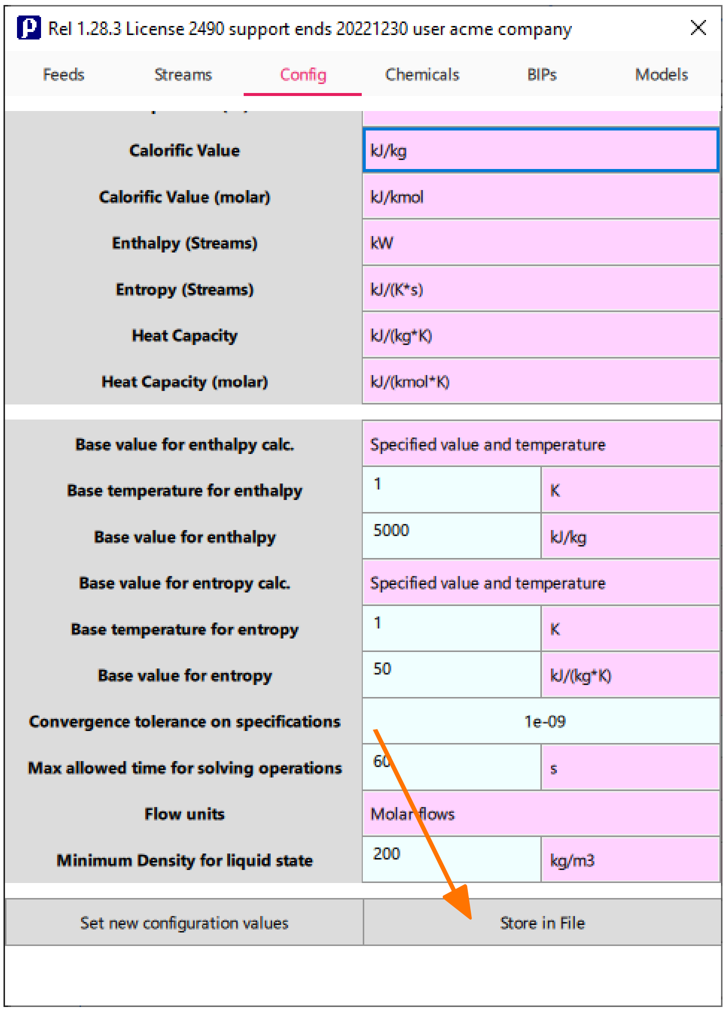
• Edit Properties : to open Prode Editor

Working with archives

Prode Properties stores data in different files

Chemical's data	: chema.dat, chemb.dat
Pseudo-component's data	: pseudo.dat
Binary Interaction parameter's data	: bips.dat
Model's data	: mod.dat
Feeds, Units of measurement, Configuration data	: def.ppp

Prode Properties editor allows to inspect and modify these archives, the different pages include buttons to overwrite these files, the user can modify existing values, add new components etc.
For example, it is possible to edit / modify / add new Feeds, Units of Measurement, Configuration parameters and then store the new values in a file so that the new information will not be lost when the user ends the program :


Rel 1.28.3 License 2490 support ends 20221230 user acme company
Feeds
Streams
Config
Chemicals
BIPs
Models
Set new configuration values
Store in File

Calorific Value	kJ/kg	
Calorific Value (molar)	kJ/kmol	
Enthalpy (Streams)	kW	
Entropy (Streams)	kJ/(K*s)	
Heat Capacity	kJ/(kg*K)	
Heat Capacity (molar)	kJ/(kmol*K)	
Base value for enthalpy calc.	Specified value and temperature	
Base temperature for enthalpy	1	K
Base value for enthalpy	5000	kJ/kg
Base value for entropy calc.	Specified value and temperature	
Base temperature for entropy	1	K
Base value for entropy	50	kJ/(kg*K)
Convergence tolerance on specifications	1e-09	
Max allowed time for solving operations	60	s
Flow units	Molar flows	
Minimum Density for liquid state	200	kg/m3

Getting Started from Microsoft Excel (predefined examples)

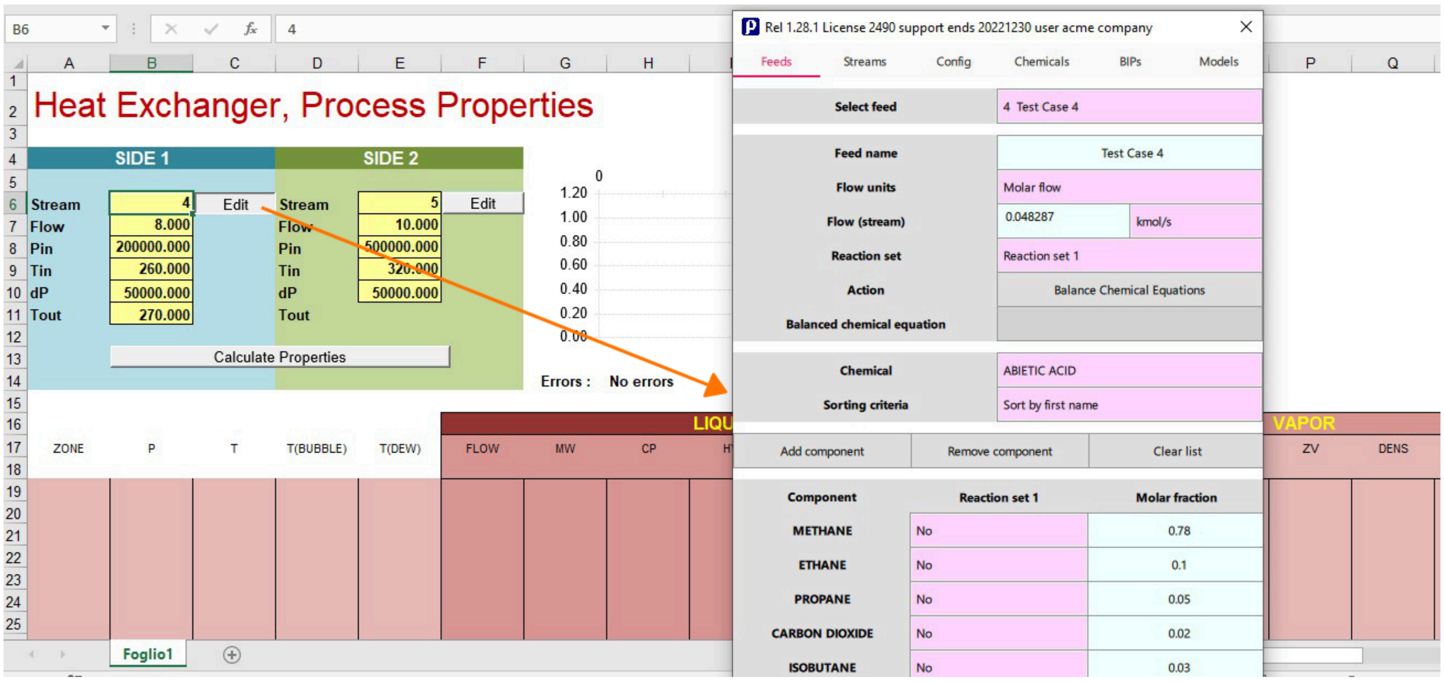
Prode Properties distribution includes several Excel examples to show how the software can solve a series of common problems, Excel support files are located in Prode/Excel folder.

These pages include Excel VBA code accessing Prode library, you can inspect and edit / modify the code with Excel developer tools, in the same way you can create your own custom pages.

Note : in these pages do not enter (in Excel cells) macros accessing Prode library to avoid conflicts and errors such as Excel not responding, if you wish to insert macros in Excel cells follow the procedure discussed in Getting Started from Microsoft Excel (working with macros), for the same reasons do not open predefined pages and pages Excel macros

This example shows how to utilize a predefined page to calculate process properties for both sides of a heat exchanger.

From Excel open the file htcprops.xls



Heat Exchanger, Process Properties

	SIDE 1			SIDE 2	
Stream	4	Edit	Stream	5	Edit
Flow	8.000		Flow	10.000	
Pin	200000.000		Pin	500000.000	
Tin	260.000		Tin	320.000	
dP	50000.000		dP	50000.000	
Tout	270.000		Tout		

Calculate Properties

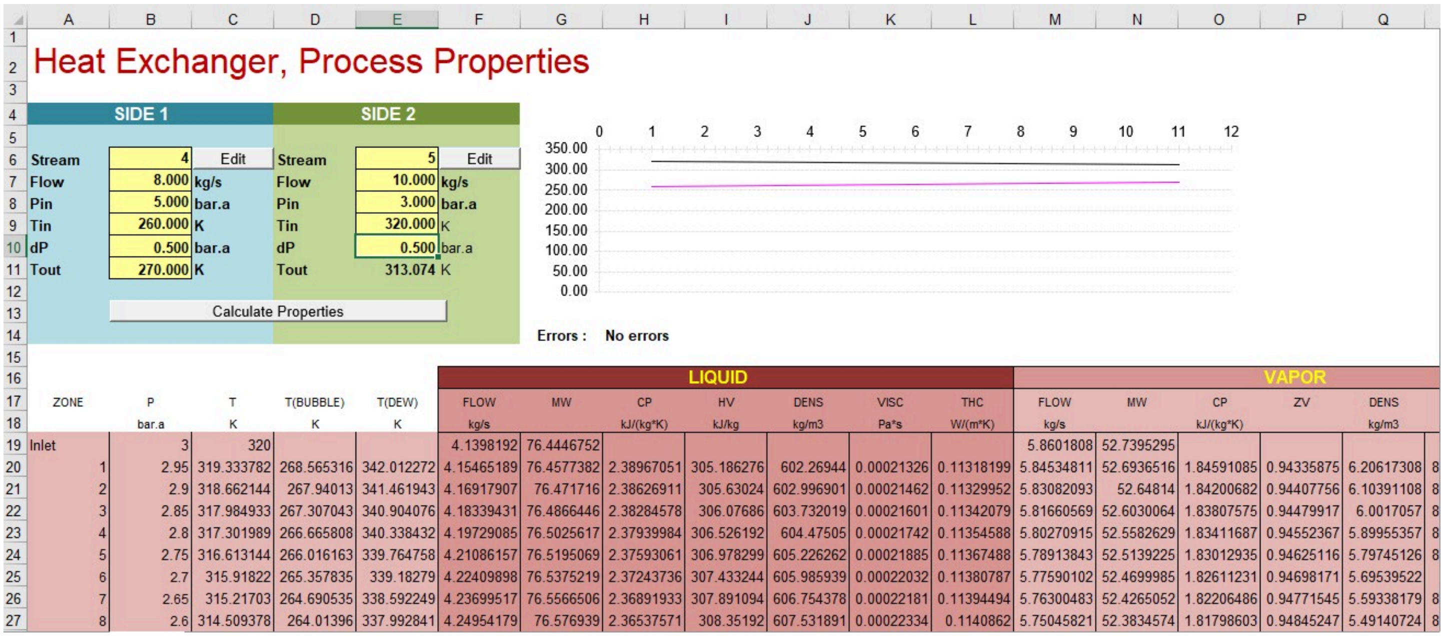
Errors : No errors

Rel 1.28.1 License 2490 support ends 20221230 user acme company

Select feed	4 Test Case 4
Feed name	Test Case 4
Flow units	Molar flow
Flow (stream)	0.048287 kmol/s
Reaction set	Reaction set 1
Action	Balance Chemical Equations
Balanced chemical equation	
Chemical	ABIETIC ACID
Sorting criteria	Sort by first name

Component	Reaction set 1	Molar fraction
METHANE	No	0.78
ETHANE	No	0.1
PROPANE	No	0.05
CARBON DIOXIDE	No	0.02
ISOBUTANE	No	0.03

use Prode Editor to inspect / edit both hot and cold streams, define the units of measurement, settings etc.



Heat Exchanger, Process Properties

	SIDE 1				SIDE 2		
Stream	4	Edit		Stream	5	Edit	
Flow	8.000	kg/s		Flow	10.000	kg/s	
Pin	5.000	bar.a		Pin	3.000	bar.a	
Tin	260.000	K		Tin	320.000	K	
dP	0.500	bar.a		dP	0.500	bar.a	
Tout	270.000	K		Tout	313.074	K	

Calculate Properties

Errors : No errors

					LIQUID							VAPOR				
ZONE	P	T	T(BUBBLE)	T(DEW)	FLOW	MW	CP	HV	DENS	VISC	THC	FLOW	MW	CP	ZV	DENS
	bar.a	K	K	K	kg/s		kJ/(kg*K)	kJ/kg	kg/m3	Pa*s	W/(m*K)	kg/s		kJ/(kg*K)		kg/m3
Inlet	3	320			4.1398192	76.4446752						5.8601808	52.7395295			
1	2.95	319.333782	268.565316	342.012272	4.15465189	76.4577382	2.38967051	305.186276	602.26944	0.00021326	0.11318199	5.84534811	52.6936516	1.84591085	0.94335875	6.20617308
2	2.9	318.662144	267.94013	341.461943	4.16917907	76.471716	2.38626911	305.63024	602.996901	0.00021462	0.11329952	5.83082093	52.64814	1.84200682	0.94407756	6.10391108
3	2.85	317.984933	267.307043	340.904076	4.18339431	76.4866446	2.38284578	306.07686	603.732019	0.00021601	0.11342079	5.81660569	52.6030064	1.83807575	0.94479917	6.0017057
4	2.8	317.301989	266.665808	340.338432	4.19729085	76.5025617	2.37939984	306.526192	604.47505	0.00021742	0.11354588	5.80270915	52.5582629	1.83411687	0.94552367	5.89955357
5	2.75	316.613144	266.016163	339.764758	4.21086157	76.5195069	2.37593061	306.978299	605.226262	0.00021885	0.11367488	5.78913843	52.5139225	1.83012935	0.94625116	5.79745126
6	2.7	315.91822	265.357835	339.18279	4.22409898	76.5375219	2.37243736	307.433244	605.985939	0.00022032	0.11380787	5.77590102	52.4699985	1.82611231	0.94698171	5.69539522
7	2.65	315.21703	264.690535	338.592249	4.23699517	76.5566506	2.36891933	307.891094	606.754378	0.00022181	0.11394494	5.76300483	52.4265052	1.82206486	0.94771545	5.59338179
8	2.6	314.509378	264.01396	337.992841	4.24954179	76.576939	2.36537571	308.35192	607.531891	0.00022334	0.1140862	5.75045821	52.3834574	1.81798603	0.94845247	5.49140724

click button “Calculate Properties” to calculate temperatures and fluid properties in the different zones of heat exchanger

the support files provided for Excel include pages to calculate phase diagrams, compressors, distillation columns, hydrate formation conditions etc.

You can contact Prode for additional application examples.

A word of warning

when open, Prode Editor prevents Excel to process user inputs (including operations on predefined pages), close Prode Editor before to access any command in Excel (to avoid possible instabilities and errors).

Getting Started from Microsoft Excel (working with macros)

Open a new Excel page, <u>to avoid conflicts and errors such as Excel not responding do not open / run the predefined pages when you utilize macros in Excel</u>.

Working with Excel you can utilize Prode Editor to edit / define streams and units of measurement, this example utilizes the predefined stream 1 (Methane 0.7, Carbon Dioxide 0.15, Hydrogen Sulfide 0.15) with units Kelvin for temperature and Bar.a for pressure.

The example shows how to calculate different properties directly in Excel, we utilize the methods discussed in paragraph "Extended methods for accessing stream's properties", these methods allows to calculate properties at specified conditions, you may wish to read the paragraph for additional information.

In B1 we enter 230 as temperature (remember we have K as unit) and in B2 we enter 25 as pressure (remember we have Bar.a as unit), the units of calculated values are Kg/m3 for density, and Kj Kg / K for heat capacity

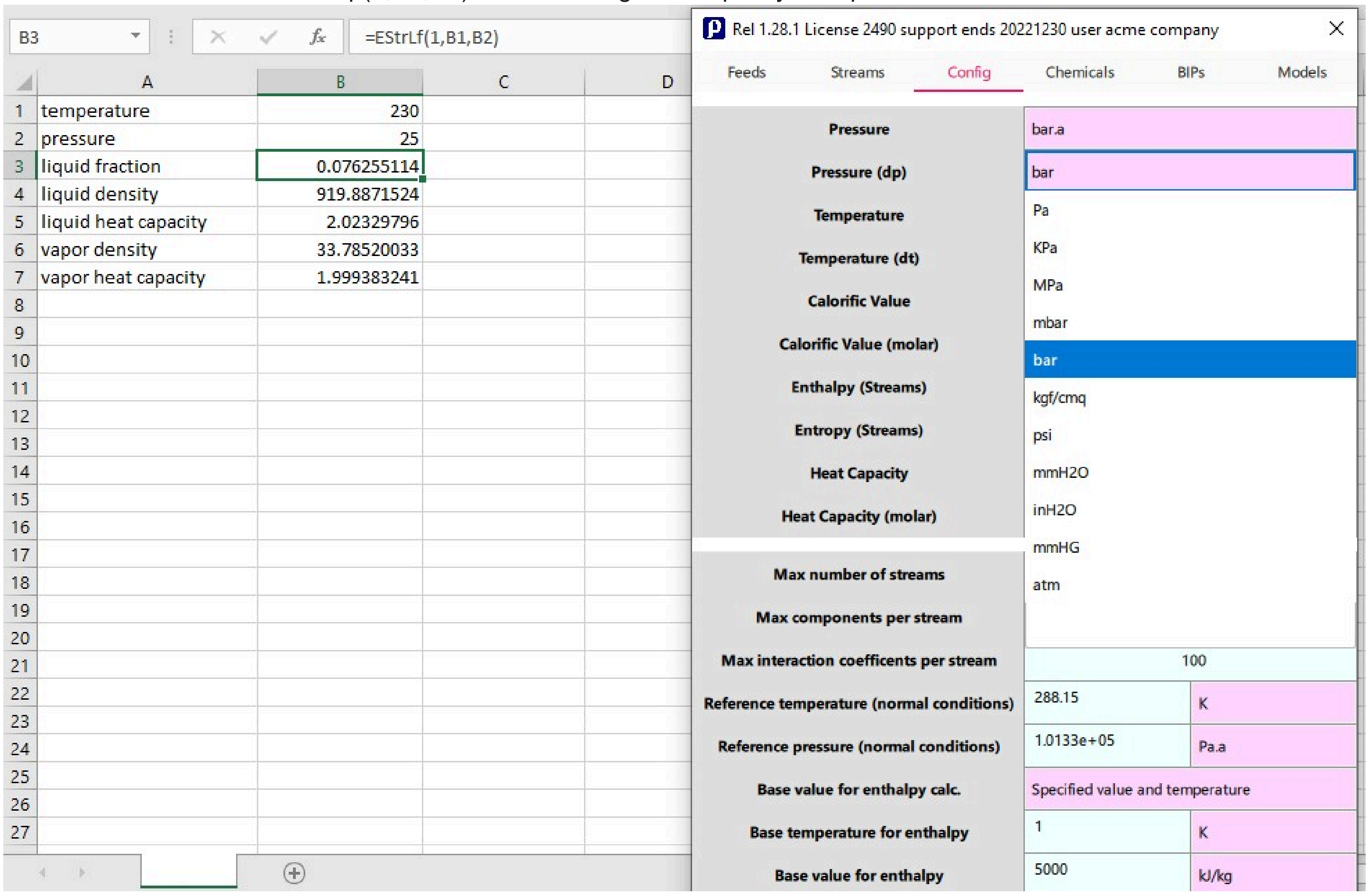
in B3 enter the macro =EStrLf(8,B1,B2) for calculating liquid fraction of stream 8 at temperature specified in B1 and pressure specified in B2

in B4 enter the macro =EStrLD(8,B1,B2) for calculating density of liquid fraction,

in B5 enter the macro =EStrLcp(8,B1,B2) for calculating heat capacity of liquid fraction,

in B6 enter the macro =EStrGD(8,B1,B2) for calculating density of vapor fraction,

in B7 enter the macro =EStrGcp(8,B1,B2) for calculating heat capacity of vapor fraction.



B3 =EStrLf(1,B1,B2)

	A	B
1	temperature	230
2	pressure	25
3	liquid fraction	0.076255114
4	liquid density	919.8871524
5	liquid heat capacity	2.02329796
6	vapor density	33.78520033
7	vapor heat capacity	1.999383241

Rel 1.28.1 License 2490 support ends 20221230 user acme company

Pressure	bar.a	
Pressure (dp)	bar	
Max interaction coefficents per stream	100	
Reference temperature (normal conditions)	288.15	K
Reference pressure (normal conditions)	1.0133e+05	Pa.a
Base value for enthalpy calc.	Specified value and temperature	
Base temperature for enthalpy	1	K
Base value for enthalpy	5000	kJ/kg

In addition to the specific methods discussed in paragraph "Extended methods for accessing stream's properties", with Excel you can utilize all the methods exported by Prode Properties library, the list includes methods to define streams, calculate a complete set of properties and solve complex operations such as columns, reactors etc.

For exaple, you can set 150 K and 5 bar.a as operating conditions in stream 1 with the macro

=setOp(1,150,5)

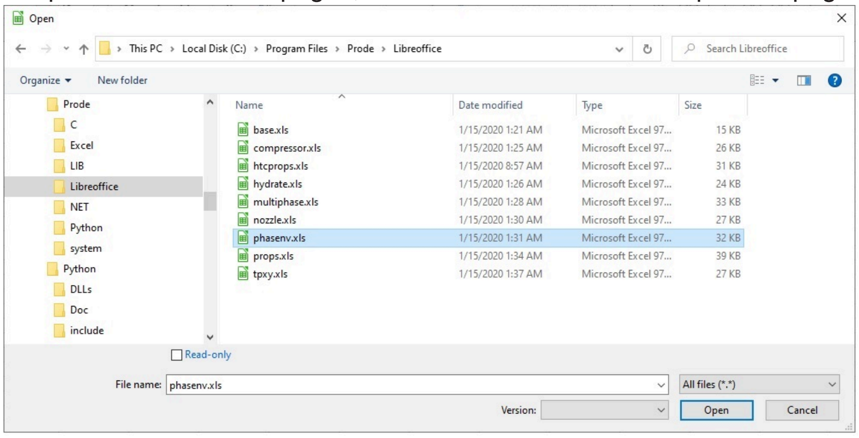
in the same way you can, for example, simulate a heat exchanger (100 KW) by calculating the enthalpy of a stream to define the new operating conditions as the result of a H-P operation, where you specify 5 Bar.a as final pressure and initial enthalpy + 100 KW

= HPF(1,5,StrH(1)+100,0)

Getting started from LibreOffice

LibreOffice (and OpenOffice) Calc tools provide many fundamental features of Excel and they include Apache Open Office Basic, a programming language similar to Microsoft Excel VBA, Prode distribution includes several LibreOffice pages in folder /Prode/LibreOffice, the LibreOffice pages look (and work) not much differently from equivalent Excel versions,

to open the LibreOffice pages, start LibreOffice Calc and Open the page phasenv.xls



Name	Date modified	Type	Size
base.xls	1/15/2020 1:21 AM	Microsoft Excel 97...	15 KB
compressor.xls	1/15/2020 1:25 AM	Microsoft Excel 97...	26 KB
htcprops.xls	1/15/2020 8:57 AM	Microsoft Excel 97...	31 KB
hydrate.xls	1/15/2020 1:26 AM	Microsoft Excel 97...	24 KB
multiphase.xls	1/15/2020 1:28 AM	Microsoft Excel 97...	33 KB
nozzle.xls	1/15/2020 1:30 AM	Microsoft Excel 97...	27 KB
phasenv.xls	1/15/2020 1:31 AM	Microsoft Excel 97...	32 KB
props.xls	1/15/2020 1:34 AM	Microsoft Excel 97...	39 KB
tpxy.xls	1/15/2020 1:37 AM	Microsoft Excel 97...	27 KB

File name: phasenv.xls

you may receive a message "LibreOffice Security Warning" : The document contains document macros , click Enable Macros button and proceed

the page includes several buttons

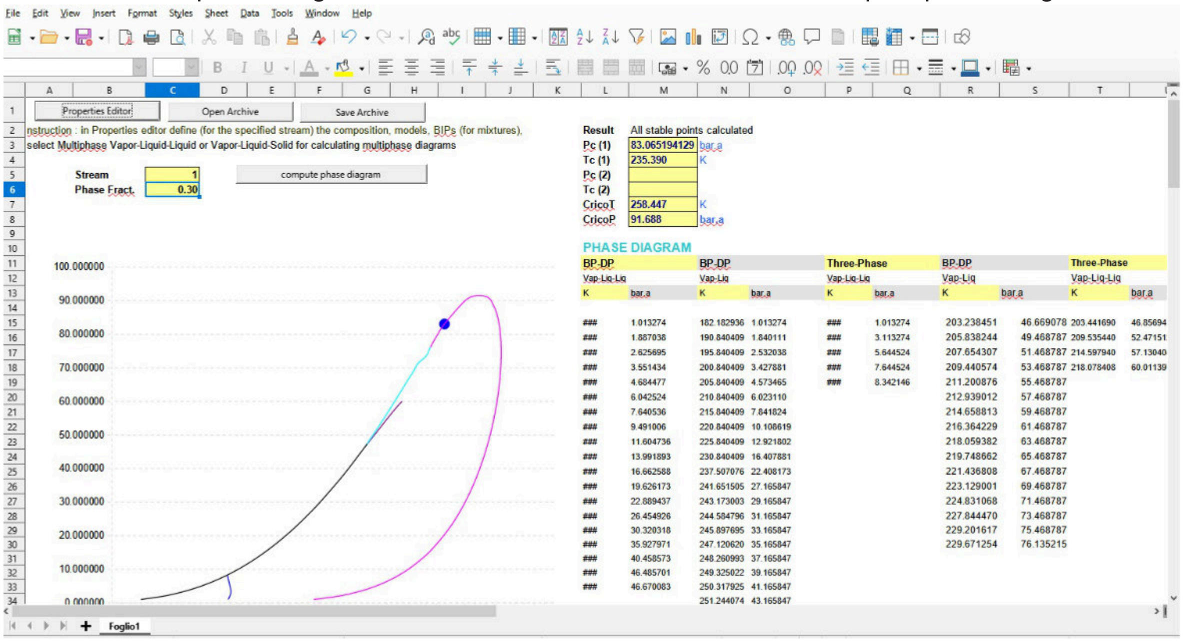
• Properties Editor opens the editor dialog

• Open Archive opens a archive

• Save Archive saves a archive

• Compute phase diagram calculates the phase diagram for the specified stream

To calculate the phase diagram define a stream and click the button Compute phase diagram



nstruction : in Properties editor define (for the specified stream) the composition, models, BIPs (for mixtures),
select Multiphase Vapor-Liquid-Liquid or Vapor-Liquid-Solid for calculating multiphase diagrams

Stream	1
Phase Fract.	0.30

Result	All stable points calculated	
Pc (1)	83.065194129	bar,a
Tc (1)	235.390	K
Pc (2)		
Tc (2)		
CricoT	258.447	K
CricoP	91.688	bar,a

PHASE DIAGRAM

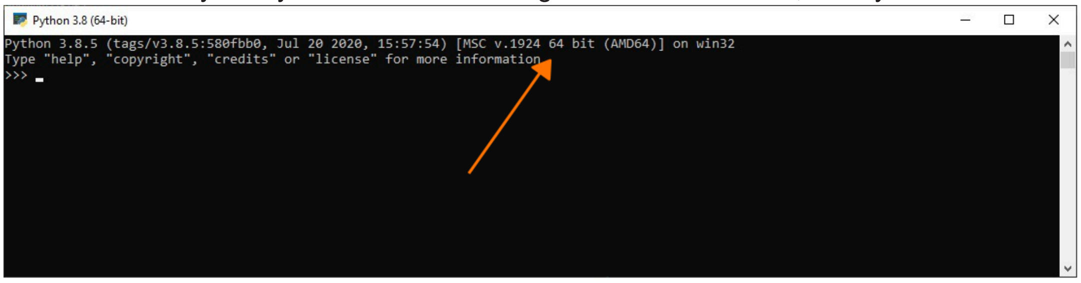
BP-DP		BP-DP		Three-Phase		BP-DP		Three-Phase	
Vap-Liq-Liq		Vap-Liq		Vap-Liq-Liq		Vap-Liq		Vap-Liq-Liq	
K	bar,a	K	bar,a	K	bar,a	K	bar,a	K	bar,a
###	1.013274	182.182936	1.013274	###	1.013274	203.238451	46.669078	203.441690	46.85694
###	1.887038	190.840409	1.840111	###	3.113274	205.838244	49.468787	209.535440	52.47151
###	2.625695	195.840409	2.532038	###	5.644524	207.654307	51.468787	214.597940	57.13040
###	3.551434	200.840409	3.427881	###	7.644524	209.440574	53.468787	218.078408	60.01139
###	4.684477	205.840409	4.573465	###	8.342146	211.200876	55.468787		
###	6.042524	210.840409	6.023110			212.939012	57.468787		
###	7.640536	215.840409	7.841824			214.658813	59.468787		
###	9.491006	220.840409	10.108619			216.364229	61.468787		
###	11.604736	225.840409	12.921802			218.059382	63.468787		
###	13.991893	230.840409	16.407881			219.748662	65.468787		
###	16.662588	237.507076	22.408173			221.436808	67.468787		
###	19.626173	241.651505	27.165847			223.129001	69.468787		
###	22.889437	243.173003	29.165847			224.831068	71.468787		
###	26.454926	244.584796	31.165847			227.844470	73.468787		
###	30.320318	245.897695	33.165847			229.201617	75.468787		
###	35.927971	247.120620	35.165847			229.671254	76.135215		
###	40.458573	248.260993	37.165847						
###	46.485701	249.325022	39.165847						
###	46.670083	250.317925	41.165847						
		251.244074	43.165847						

in the same way you can load the pages for solving different problems as discussed in Excel section

Caution : the last versions of LibreOffice Calc can result unstable when loading large external libraries such as Prode Editor, opening / closing Prode Editor and solving worksheets without pausing between operations you may experience occasional crashes, feel free to contact Prode for specific information and support.

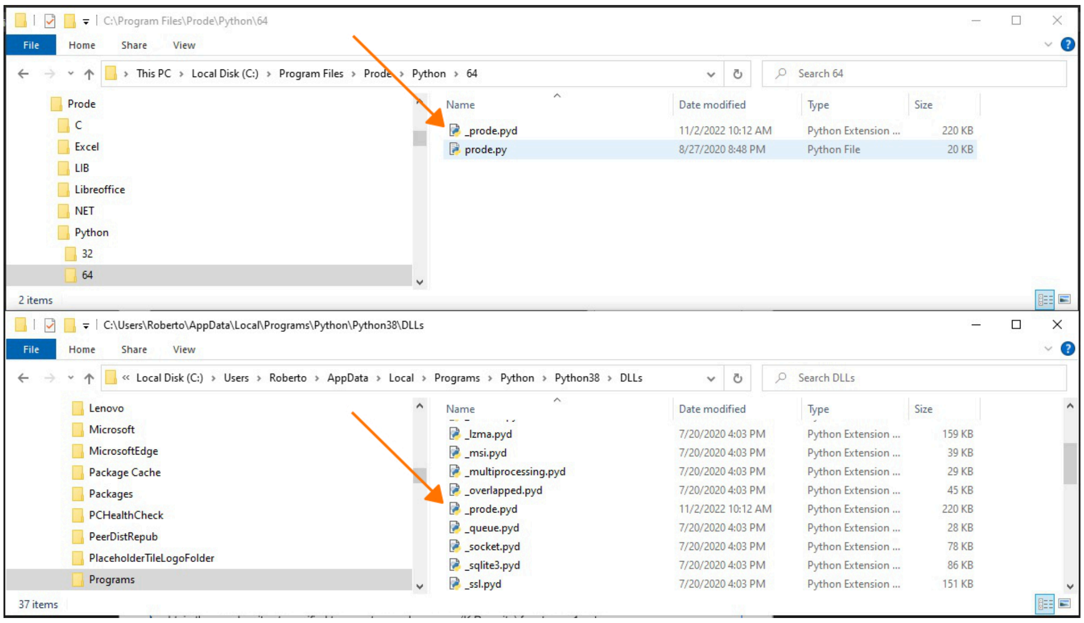
Getting started from Python

The different versions of Prode Properties work with the different versions of Python available for Windows, Linux, Android platforms and many methods exposed by Prode library can be imported in Python applications,
in Windows, to install Prode Properties library in your Python application, follow these steps
• check whether your Python shell is executing in 32 or 64 bit mode, run Python and read the data (this is 64 bit)



```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

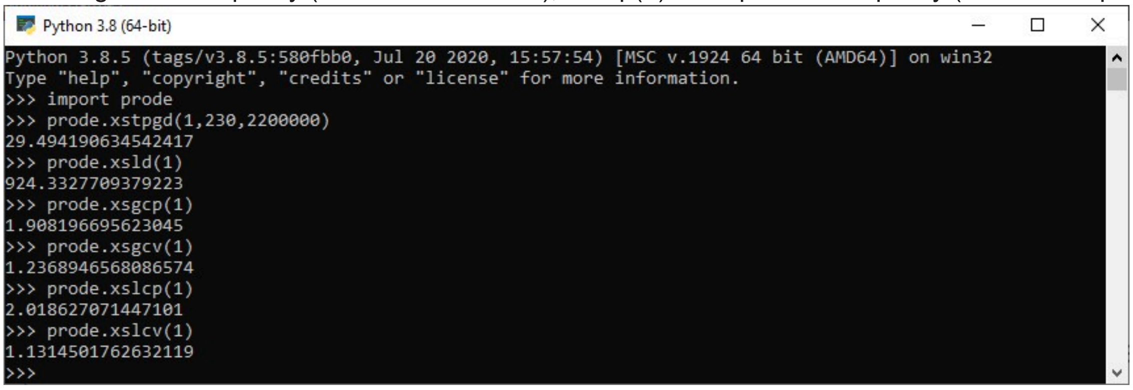
for a 64 bit Python shell copy the files from Prode\Python\64 folder, for a 32 bit from Prode\Python\32 folder
• copy prode.py to your Python install in /Lib folder


C:\Program Files\Prode\Python\64
File
Home
Share
View
This PC › Local Disk (C:) › Program Files › Prode › Python › 64
Search 64
Prode
C
Excel
LIB
Libreoffice
NET
Python
32
64
2 items
C:\Users\Roberto\AppData\Local\Programs\Python\Python38\DLLs
« Local Disk (C:) › Users › Roberto › AppData › Local › Programs › Python › Python38 › DLLs
Search DLLs
Lenovo
Microsoft
MicrosoftEdge
Package Cache
Packages
PCHealthCheck
PeerDistRepub
PlaceholderTileLogoFolder
Programs
37 items

Name	Date modified	Type	Size
_prode.pyd	11/2/2022 10:12 AM	Python Extension ...	220 KB
prode.py	8/27/2020 8:48 PM	Python File	20 KB

Name	Date modified	Type	Size
_lzma.pyd	7/20/2020 4:03 PM	Python Extension ...	159 KB
_msi.pyd	7/20/2020 4:03 PM	Python Extension ...	39 KB
_multiprocessing.pyd	7/20/2020 4:03 PM	Python Extension ...	29 KB
_overlapped.pyd	7/20/2020 4:03 PM	Python Extension ...	45 KB
_prode.pyd	11/2/2022 10:12 AM	Python Extension ...	220 KB
_queue.pyd	7/20/2020 4:03 PM	Python Extension ...	28 KB
_socket.pyd	7/20/2020 4:03 PM	Python Extension ...	78 KB
_sqlite3.pyd	7/20/2020 4:03 PM	Python Extension ...	86 KB
_ssl.pyd	7/20/2020 4:03 PM	Python Extension ...	151 KB

• Copy prode.pyd to your Python install in /DLLs folder
run Python shell, to import prode module in Python, type
>>> import prode
then you have access to many methods exposed by Prode library, for example to obtain the gas density at 230 K and 2200000 Pa (22 Bar.a) for stream 1 enter
>>> prode.xstpgd(1,230,2200000)
xstpgd works as EstrGD(), the names to utilize with Python are those indicated in the documentation with the prefix sname
>>> prode.xsgcv(1)
returns gas heat capacity (at constant volume), xslcp(1) the liquid heat capacity (at constant pressure) etc.



```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import prode
>>> prode.xstpgd(1,230,2200000)
29.494190634542417
>>> prode.xsld(1)
924.3327709379223
>>> prode.xsgcp(1)
1.908196695623045
>>> prode.xsgcv(1)
1.2368946568086574
>>> prode.xslcp(1)
2.018627071447101
>>> prode.xslcv(1)
1.1314501762632119
>>>
```

with Python you can create complex procedures, feel free to contact Prode for additional information

Accessing Prode Properties library (with programming languages)

The technique for accessing the methods in Prode Properties library will depend on which programming language you use. Languages such as FORTRAN, C, C++ or Microsoft NET (VB,C) exhibit differences in parameter passing in and out of functions. This may require you to adapt your code from the examples shown here. The calling convention determines how a program makes a call and where the parameters are passed.

Prode Properties does use of standard calls, it pushes parameters on the stack, in reverse order. When accessing Properties consider :

• Prode Properties real (double) type is 8 bytes

• Prode Properties integer type is 4 bytes

• parameters are passed by value (with exception of strings which are arrays of characters)

C / C++

• include the ppp.h, pppx.h headers

• add ppp.lib, pppx.lib files to the list of the files in your project

• make sure you use the calling convention of ppp.h header file,

• from your code call the methods in Prode Properties library

• feel free to contact Prode for additional information and support

Fortran

add ppp.lib file to the list of the files in project and include ppp.f90 to instruct the compiler about the methods available in Prode Properties then access the methods as they were included in your code

```
C this procedure returns the critical temperature of a compound
INTERFACE TO REAL*8 FUNCTION TC ([C,ALIAS:'CompTc'] comp)
INTEGER*4 comp [VALUE]
END
REAL*8 tc
INTEGER*4 id
C define the id value here
tc = TC(id)
```

Microsoft NET

we can provide samples for C# and VB#, feel free to contact us for additional information and support

Microsoft Excel (VBA)

see the samples provided, feel free to contact Prode for additional information and support

OpenOffice

see the samples provided, feel free to contact Prode for additional information and support

Python

At present we do not include samples but feel free to contact us for additional information and support

Some tips on creation of Prode Properties applications

• include access to Properties Editor, for example with method edSS() to simplify debug operations, when debugging always attempt to limit the complexity of problems and expand progressively to the full application, retesting at intervals as you expand the scope of your problem.

• ensure that units of measurement are correct / include methods to set the units.

• utilize isSDef() method to test a streams validity before accessing the stream, accessing undefined streams generates a large numbers of errors.

• utilize methods / procedures to test errors on each step, specifically for long calculation sequences.

Introducing Prode Properties library methods

Prode Properties library includes a range of methods to deal with problems in chemical engineering and to achieve tight control over the calculations .

A non-inclusive list would include

- Thermodynamic calcs (flash operations, enthalpy, entropy, volume, energy, unit operations)
- Streams data access and calcs (set and retrieve operating conditions, critical and transport properties calcs)
- Chemicals library access (retrieve data from chemicals file)
- Error messages (management of errors messages)

Methods for thermodynamic calc' s

Prode Properties includes a complete set of methods for solving all the standard flash operations with specified final temperature or pressure and entropy or enthalpy or volume or energy basis, phase fraction with temperature or pressure basis plus mixers, dividers, gas,liquid phase separation operations etc.

integer result = setOp(integer stream, double t, double p)
sname xftp
Given a stream, operating pressure and temperature, performs an isothermal flash and sets operating conditions.

integer result = setSOp(integer stream)
sname xfstp
Given a stream performs an isothermal flash at (user defined) standard conditions.

double t = PfPF(integer stream, double p, double pf, int state, int n)
Sname xfpfp
Given a stream, the pressure , phase fraction (range 0-1), state (gas, liquid, solid) and position n calculates and returns the nth (n : 1-5) equilibrium temperature along the specified phase fraction line

double p = PfTF(integer stream, double t, double pf, int state, int n)
sname xfpft
Given a stream, the temperature , phase fraction (range 0-1), state (gas, liquid, solid) and position n calculates and returns the nth (n : 1-5) equilibrium pressure along the specified phase fraction line

double t = LfPF(integer stream, double p, double lf)
sname xflfp
Given a stream, the pressure and Liquid fraction (range 0-1) calculates and returns the first equilibrium temperature along the specified phase fraction line

double p = LfTF(integer stream, double t, double lf)
sname xflft
Given a stream, the temperature and Liquid fraction (range 0-1) calculates and returns the first equilibrium pressure along the specified phase fraction line

double t = BPF(integer stream, double p)
sname xfbp
Given a stream and pressure calculates and returns bubble point temperature

double t = DPF(integer stream, double p)
sname xfdp
Given a stream and pressure calculates and returns dew point temperature

double t = HPF(integer stream, double p, double h, double et)
sname xfhp
Given a stream, final pressure, the required (final) enthalpy (see the method StrH() for the definition) and a estimated value for final temperature (or 0 for automatic estimate), method solves the flash operation (enthalpy basis) and returns final temperature

double p = HTF(integer stream, double t, double h, double ep)
sname xfht
Given a stream, final temperature, the required (final) enthalpy (see the method StrH() for the definition) and a estimated value for final pressure (or 0 for automatic estimate), method solves the flash operation (enthalpy basis) and returns final pressure

double t = SPF(integer stream, double p, double s, double et)

sname xfsp

Given a stream, final pressure, the required (final) entropy (see the method StrS() for the definition) and a estimated value for final temperature (or 0 for automatic estimate), method solves the flash operation (entropy basis) and returns final temperature.

double p = STF(integer stream, double t, double s, double ep)

sname xfst

Given a stream, final temperature, the required (final) entropy (see the method StrS() for the definition) and a estimated value for final pressure (or 0 for automatic estimate), method solves the flash operation (entropy basis) and returns final pressure.

double t = VPF(integer stream, double p, double v, double et)

sname xfvp

Given a stream, final pressure, the required specific volume (see the method StrV() for the definition) and a estimated value for final temperature (or 0 for automatic estimate), method solves the flash operation (volume basis) and returns final temperature.

double p = VTF(integer stream, double t, double v, double ep)

sname xfvt

Given a stream, final temperature, the required specific volume (see the method StrV() for the definition) and a estimated value for final pressure (or 0 for automatic estimate), method solves the flash operation (volume basis) and returns final pressure.

integer result = HVF(integer stream, double h, double v, double et, double ep)

sname xfhv

Given a stream, the required (final) enthalpy (see the method StrH() for the definition) the required (final) specific volume (see the method StrV() for the definition) and estimated values for final temperature and pressure (or 0 for automatic estimate), method solves the flash operation

integer result = SVF(integer stream, double s, double v, double et, double ep)

sname xfsv

Given a stream, the required (final) entropy (see the method StrS() for the definition) the required specific volume (see the method StrV() for the definition) and estimated values for final temperature and pressure (or 0 for automatic estimate), method solves the flash operation

integer result = HSF(integer stream, double h, double s, double et, double ep)

sname xfhs

Given a stream, the required (final) enthalpy (see the method StrH() for the definition) the required (final) entropy (see the method StrS() for the definition) and estimated values for final temperature and pressure (or 0 for automatic estimate), method solves the flash operation

double t = EPF(integer stream, double p, double E, double aout, double et)

sname xfep

Given a stream, final pressure, outlet area, the term E (equal to Hin + 1/2Vin^2) and a estimated value for final temperaure (or 0 for automatic estimate) method solves the constant energy flash and returns final temperature, method solves Hin + 1/2Vin^2 = Ho + 1/2Vo^2 and it permits to model adiabatic, irreversible expansions when the contribute of kinetic energy cannot be neglected.

integer result = MixF(integer stream1, integer stream2, double et)

sname xmix

Given two streams, stream1 and stream2 and a estimated value for final temperature (or 0 for automatic estimate) method solves a mixer operation and returns the result on stream1, the feed streams are adiabatically flashed to the lowest inlet stream pressure

integer result = Divi (integer stream1, integer stream2, double wdiv)

sname xdivi

Given two streams (stream1 and stream2) and a flowrate fraction (0-1) performs a divider operation so that stream 1 is shifted into two streams (stream1, stream2) of the same composition, temperature and pressure, flowrate fractions are subdivided as specified by wdiv (stream2 = wdiv, stream1 = 1- wdiv)

integer result = psep(integer stream1, integer stream2, integer phase)
sname xpsep
Given a stream (stream1) performs an isothermal flash to simulate a phase type (vapor,liquid,solid) separator and returns the result as stream2.

integer res = StrCopy(integer stream1, integer stream2)
sname xscopy
Given two streams (stream1 and stream2) copies the stream 2 into stream 1, the method copies all valid data including operating data if available.

Methods for stream' s data access

Prode Properties includes a set of functions for accessing stream parameters and calculating transport properties.

integer res = isSDef(integer stream)
sname xsdef
given a stream returns TRUE (value = 1) if stream has been defined, otherwise returns FALSE (0)

double t = getT(integer stream)
sname xst
given a stream returns stream's operating temperature

double p = getP(integer stream)
sname xsp
given a stream returns stream's operating pressure

integer nr = getPNr()
sname xpnr
returns the maximum number of phases that procedure can detect

integer type = StrPt(integer stream, int phase)
sname xspt
given a stream and position in range 1- getPNr() returns the phase type (vapor,liquid,solid)

char *description = StrPts(integer stream, int phase)
sname xspts
given a stream and position in range 1- getPNr() returns a ANSI C string with the description (vapor, liquid, solid...)

int description MStrPts(integer stream, int phase, char *s, integer slm)
given a stream and position in range 1- getPNr() fills string s with the description (vapor, liquid, solid...) (eventually truncated to slm maximum lenght), this is the Microsoft Excel specific method

double lf = StrLf(integer stream)
sname xslf
given a stream returns the total liquid fraction (molar basis) in stream

double pf = StrPf(integer stream, integer phase)
sname xspf
given a stream and phase position in range 1- getPNr() returns the phase fraction

double w = getW(integer stream, integer phase, integer pos.)
given a stream, the phase position and component's position (in component's list) returns the component molar fraction in that phase

double Zi= getZ(integer stream, integer pos.)
sname xsz
given a stream and component's position (in component's list) returns the comp's Z (weight percentage, molar basis)

integer res = putZ(integer stream, integer pos., double Zi)
sname xsetsz
given a stream, comp's position and Z , sets the comp's pos. in Z vector (composition, molar basis) for that stream

integer nr = getCNr(integer stream)
sname xscnr
given a stream returns the number of components defined in that stream

integer nr = getMCNr()
sname xsmcnr
returns the maximum number of components in a stream

double zv = StrZv(integer stream)
sname xszv
given a stream returns the relevant compressibility factor (gas phase)

double mw = StrMw(integer stream)
sname xsmw
given a stream returns the averaged molecular weight (all phases)

double v = StrV(integer stream)
sname xsv
given a stream returns the specific volume as sum of specific volumes of all phases

double mw = StrGMw(integer stream)
sname xsgmw
given a stream returns the averaged molecular weight (gas phase)

double mw = StrLMw(integer stream)
sname xslmw
given a stream returns the averaged molecular weight (liquid phase)

double h = StrH(integer stream)
sname xsh
given a stream returns the total (stream) enthalpy (gas + liquid + solid phases)

double h = StrGH(integer stream)
sname xsgh
given a stream returns the total (stream) enthalpy (gas phase)

double h = StrSGH(integer stream)
sname xssgh
given a stream returns the specific (unit weight) enthalpy (gas phase)

double h = StrLH(integer stream)
sname xslh
given a stream returns the total (stream) enthalpy (liquid phase)

double h = StrSLH(integer stream)
sname xsslh
given a stream returns the specific (unit weight) enthalpy (liquid phase)

double h = StrSH(integer stream)
sname xssh
given a stream returns the total (stream) enthalpy (solid phase)

double h = StrSSH(integer stream)
sname xsssh
given a stream returns the specific (unit weight) enthalpy (solid phase)

double cp = StrGICp(integer stream)
sname xsgicp
given a stream returns the ideal gas heat capacity

double cp = StrGCp(integer stream)
sname xsgcp
given a stream returns the specific heat capacity (constant pressure, gas phase)

double cv = StrGCv(integer stream)
sname xsgcv
given a stream returns the specific heat capacity (constant volume, gas phase)

double cp = StrLCp(integer stream)
sname xslcp
given a stream returns the specific heat capacity (constant pressure, liquid phase)

double cv = StrLCv(integer stream)
sname xslcv
given a stream returns the specific heat capacity (constant volume, liquid phase)

double cp = StrSCp(integer stream)
sname xsgcp
given a stream returns the specific heat capacity (constant pressure, solid phase)

double ss = StrMSS(integer stream)
sname xsmss
given a stream returns the speed of sound (gas, liquid) as calculated with HEM model for mixed phases

double ss = StrGSS(integer stream)
sname xsgss
given a stream returns the speed of sound in gas phase

double ss = StrLSS(integer stream)
sname xlmss
given a stream returns the speed of sound in liquid phase

double jt = StrGJT(integer stream)
sname xsgjt
given a stream returns the Joule Thomson coefficient in gas phase

double jt = StrLJT(integer stream)
sname xsljt
given a stream returns the Joule Thomson coefficient in liquid phase

double ic = StrGIC(integer stream)
sname xsgic
given a stream returns the isothermal compressibility coefficient - (1 / V) * dV / dP in gas phase

double ic = StrLIC(integer stream)
sname xslic
given a stream returns the isothermal compressibility coefficient - (1 / V) * dV / dP in liquid phase

double v = StrGVE(integer stream)
sname xsgve
given a stream returns the volumetric expansivity coefficient - (1 / V) * dV / dT in gas phase

double ic = StrLVE(integer stream)
sname xslve
given a stream returns the volumetric expansivity coefficient - (1 / V) * dV / dT in liquid phase

double s = StrGS(integer stream)
sname xsgs
given a stream returns the total (stream) entropy (gas phase)

double s = StrSGS(integer stream)
sname xssgs
given a stream returns the specific (unit weight) entropy (gas phase)

double s = StrLS(integer stream)
sname xsls
given a stream returns the total (stream) entropy (liquid phase)

double s = StrSS(integer stream)
sname xsss
given a stream returns the total (stream) entropy (solid phase)

double s = StrSLS(integer stream)
sname xssls
given a stream returns the specific (unit weight) entropy (liquid phase)

double s = StrSSS(integer stream)
sname xssss
given a stream returns the specific (unit weight) entropy (solid phase)

double s = StrS(integer stream)
sname xss
given a stream returns the total (stream) entropy (gas + liquid + solid phases)

integer res = setWm(integer stream, double W)
sname xsetswm
given a stream and flow (mass basis), sets the flow

double w = getWm(integer stream)
sname xswm
given a stream returns the flow specified for that stream.

double hc = StrHC(integer stream)
sname xshc
given a stream returns the calculated net heat of combustion (gas phase).

double fl = StrFML(integer stream)
sname xsfml
given a stream returns the calculated flammability lean limit (gas phase).

double fl = StrFMH(integer stream)
sname xsfmh
given a stream returns the calculated flammability rich limit (gas phase).

double d = StrLD(integer stream)
sname xsld
given a stream returns the calculated liquid density (at operating conditions)

double d = StrGD(integer stream)
sname xsgd
given a stream returns the calculated gas density (at operating conditions)

double tc = StrLC(integer stream)
sname xsgd
given a stream returns the calculated liquid thermal conductivity (at operating conditions)

double tc = StrGC(integer stream)
sname xsgc
given a stream returns the calculated gas thermal conductivity (at operating conditions)

double v = StrLV(integer stream)
sname xslv
given a stream returns the calculated liquid viscosity (at operating conditions)

double v = StrGV(stream)
sname xsgv
given a stream returns thecalculated gas viscosity (at operating conditions).

double st = StrST(integer stream)
sname xsst
given a stream returns the calculated surface tension (at operating conditions).

Integer cpnr = StrCPnr(integer stream)
sname xscpnr
given a stream returns the number of critical points detected and calculated, to get a critical point use the methods StrPc()
And StrTc() setting value of pos in the range 1-cpnr

double p = StrPc(integer stream, Integer pos)
sname xspc
given a stream and the critical point position in the list (see method StrCPnr()) returns the critical pressure

double t = StrTc(integer stream, Integer pos)
sname xstc
given a stream and the critical point position in the list (see method StrCPnr()) returns the critical temperature.

double p= StrCBp(integer stream)
sname xscbp
given a stream returns the cricodenBar pressure.

double t= StrCBt(integer stream)
sname xscbt
given a stream returns the cricodenBar temperature.

double p= StrCTp(integer stream)
sname xsctp
given a stream returns the cricodenTherm pressure.

double t= StrCTt(integer stream)
sname xsctt
given a stream returns the cricodenTherm temperature.

double ac = StrAc(integer stream)
sname xsac
given a stream returns the acentric factor (mole fraction average).

double p= StrRVP(integer stream, integer mode)
sname xsrvp
given a stream returns the Reid vapor pressure
mode = 1 simulation of D6377 procedure (liquid not saturated with air)
mode = 2 simulation of D323 procedure (liquid saturated with air)

double fp = StrFLP(integer stream)
sname xsflp
given a stream returns the Flash point (for pure fluids the method returns the value stored in databank while for mixtures the flash point is calculated by a iterative procedure where VLE is solved according the selected models for stream)

Methods to work with packages

Each package stores a set of models for fugacity, enthalpy, entropy, volume and the different states (vapor, liquid, solid, hydrate), the library includes methods to define, store and edit packages

Integer nr = getPKnr()
xname xpknr
return the max number of packages

Integer nr = getPKdnr()
xname = xpknr
return the number of packages with valid data

char *str= getPKN(int pkg)
xname = xpkn
given the package this method returns the name

integer res = putPKN(int pkg, char* name)
xname = xsetpkn
given the package and the name this method sets the name

nteger putPKM(int pkg, int prop, int state, int model)
xname = xsetpkm
given the package, property, state and model the method sets the model and return true

integer getPKM(int pkg, int prop, int state)
xname = xpkm
given the package, property and state the method returns the model

integer res = putPKS(int pkg, int option, int value)
xname = xsetpks
given the package, property, option and value the method sets the option and return true

integer getPKS(int pt, int option)
xname = xpks
given the package and option the method returns the value

Methods to work with streams

Each stream stores a list of components and molar fractions, the associated models etc. the library includes methods to define, store and edit streams

to define a stream :

- call initS()
- for each component in the list
    set the component's code with putCC()
    set the component's mole fraction with putZ()
- call setS() to define the stream
- call setW() to define the flow
- utilize the methods described in paragraph "Methods to define thermodynamic models" to define the models
- call loadSB() to load the BIPs from database or define specific BIPs with methods PutCi(), PutCj(), PutMB(), PutBIP()

or, to simply change the component's fractions :

- for each component in the list
    set the new component's mole fraction with putZ()
- call setS() to define the stream

List of methods exported

integer res = initS (integer stream)
sname xinits
given a stream initializes all data, call this method before to create a new list of components.

integer res = putCC (integer stream, integer pos, integer compcode)
sname xsetscc
given a stream, component's position (in component's list) and component code sets the code in component's list.

integer res = putZ(integer stream, integer pos., double Zi)
sname xsetsz
given a stream, comp's position and Z , sets the comp's pos. in Z vector (composition, molar basis) for that stream

integer res = setS(integer stream)
sname xsets
given a stream performs a sequence of validating operations on data. This method must be called after to have restored stream's data from archives (files etc.)Methods to define a initial condition for a stream

nteger res = loadSB(integer stream, integer btype)
sname xloadsb
given a stream loads all BIP available in database. This method must be called after the stream has been defined since it requires the list of components. Codes for btype are 0 for VLE, 1 for LLE, 2 for SLE, 3 for Hydrates

double Zi= getZ(integer stream, integer pos.)
sname xsz
given a stream and component's position (in component's list) returns the comp's Z (molar fraction)

integer cc = getCC(integer stream, integer pos)
sname xscc
given a stream and component's position (in component's list) returns the component code (a integer that identifies the component in chemical's file).

integer nr = getMBPNr()
sname xsmbnr
returns the maximum number of (interaction coefficients) binary pairs in a stream

int ci = getCi(integer stream, integer pos)
sname xsci
given a stream and position (in interaction's coeff. list) returns the first component reference (a integer that identifies the component in component's list)

integer res = PutCi (integer stream, integer pos, integer ci)
sname xsetsci
given a stream, position (in interaction coefficients list) and first component reference sets the component's reference in interaction coefficient's list.

int cj = getCj(integer stream, integer pos)
sname xscj
given a stream and position (in interaction's coeff. list) returns the second component reference (an integer that identifies the component in component's list)

integer res = PutCj (integer stream, integer pos, integer cj)
sname xsetscj
given a stream, position (in interaction coefficients list) and second component reference sets the component's reference in interaction coefficient's list

int model = getMB(integer stream, integer pos)
sname xsmb
given a stream and position (in interaction’s coeff. list) returns the related model (an integer that identifies the model).

integer res = PutMB(integer stream, integer pos, integer model)
sname xsetsmb
given a stream, position (in interaction coefficients list) and a model identifier sets the model in interaction coefficient’s list.

double BIP = getBIP(integer stream, integer pos, integer id)
sname xsbip
given a stream, position (in binary coeff. list) and BIP identifier (0-max nr. of BIPs for that model) returns BIP.

integer res = PutBIP(integer stream, integer pos, integer id. double Kji)
sname xsetsbip
given a stream, position (in binary coeff. list) BIP identifier (0-max nr. of BIPs for that model) and value stores BIP in that position of the list.

Methods to define stream’s operating conditions

Prode Properties includes a set of functions to define phase fractions, the different phase’s compositions etc. in a operating stream, these can be utilized, for example, to enter data calculated with another software

- call rstValidSop()
- for each phase
    - for each component define fraction with putW()
    - define phase fraction with putPF()
    - define phase type with putPT()
    - set phase as valid , setValidPhase()
- define temperature with putT()
- define pressure with putP()
- set conditions as valid with setValidSop()

List of methods exported

integer result = rstValidSop(integer stream)
Sname xrstvop
Given a stream clears the compostions of different phases at operating conditions

integer result = setValidSop(integer stream)
sname xsetvop
Given a stream sets the compostions of different phases at operating conditions.as valid.

integer result = setValidPhase(integer stream, integer phase)
sname xsetvphase
Given a stream and phase sets the phase compostion.as valid.

integer result = putW(integer stream, integer phase, int compnr, double w)
sname xsetw
Given a stream, phase, component number and component’s molar fraction in that phase stores the value

integer result = putPF(integer stream, integer phase, double fraction)
sname xsetpf
Given a stream, phase and phase fraction stores the phase .fraction value

integer result = putPT(integer stream, integer phase, int type)
sname xsetpt
Given a stream, phase and phase type (vapor,liquid,solid) stores the phase type

nteger result = putT(integer stream, double t)
sname xsetst
Given a stream and operating temperature stores the value

nteger result = putP(integer stream, double p)
sname xsetsp
Given a stream and operating pressure stores the value

Methods for solving staged columns

Properties includes a procedure for solving staged columns (versions for continuous and batch distillation), the column is modeled with stgnr equilibrium stages, column may include a condenser and a reboiler, stage numbering is bottom up, the bottom stage (reboiler, if specified) is number one and the top stage (condenser, if specified) is number stgnr

There may be one or more feeds, a feed is modeled by entering liquid on the specified stage and vapor portion to the stage above (with exception of top stage).

There may be one or more side streams

Heat added / removed on each stage can be specified

Efficiency parameter on each stage can be specified

integer res = DCOL(int csep, int stgnr, int init, double *stgt,double *stgp,double *stgef,double *stgdH, int prod_h, int btm_h, int fnr,int *fstr,int *fpos,int snr,int *sstr,int *spos,int *sft, double *sflow,int vnr,double *vrv,int *vtype,int *ptype,int *piv,double *prv, double*flows)

Parameters :

csep	(int)	column type : 1 VLE , 2 VLLE , 3 LLE (some features available in extended versions)
stgnr	(int)	number of stages
init	(int)	0 for automatic initialization, 1 temperatures and flows are defined by user
stgt	(double*)	vector (stgnr) with stage temperatures
stgp	(double*)	vector (stgnr) with specified stage pressures
stgef	(double*)	vector (stgnr) with specified stage efficiency, permitted range 0,1-1
stgdH	(double*)	vector (stgnr) with specified dH (heat added, removed)
prod_h	(int)	stream for top product/distillate
btm_h	(int)	stream for bottom product
fnr	(int)	number of feeds
fstr	(int*)	vector (fnr) with the feeding streams
fpos	(int*)	vector (fnr) with feeds positions 1-stgnr
snr	(int)	number of side streams
sstr	(int*)	vector (snr) with the list of side streams
spos	(int*)	vector (snr) with side streams positions (1-stgnr)
sft	(int*)	vector (snr) with specified flow type (GAS_PHASE, LIQ_PHASE, see Codes used in Prode library)
sflow	(double*)	vector (snr) with the specified (on each side stream) side product to feed flow ratio
vnr	(int)	number of variables to solve
vtype	(int*)	vector (vnr) with type of variable (seebelow)
vrv	(double*)	vector (vnr) with calculated values for variable
ptype	(int*)	vector (pnr) with type of specification (see below)
piv	(int*)	vector (pnr) with integer values as the position of components in the list
prv	(double*)	vector (pnr) with values of the specifications to solve
flows	(double*)	vector with calculated values for vapor/liquid flows in all stages, dimension nrphases*nrc*stgnr when a condenser is present the reflux is the liquid flow on top stage

Notes :

When passing / returning paramenters the first element in vectors is the element 0

Main variables (1-vnr) are (when specified) reboiler and condenser (partial or total), each variable (of type defined in vtype) requires a suitable specification (in ptype, piv, prv), usually for reboiler the specification is the product to feed ratio and for a condenser the reflux ratio, but specifications based on component's fractions on top and bottom products are permitted, in these cases specify in piv the position of selected component in the list and in prv the value of the fraction required

Secondary variables are side streams (1-snr), each side stream (defined in sstr, spos) requires (in sflow) a specification for the side product to (total) feed flow ratio.

The column is modeled with thermodynamics and options defined for the first feed in the list.

Initialization

in most cases the procedure doesn't require to initialize values, when required set the variable init to 1 and define the initial values in vectors stgt and flows, note that in a sequence of similar operations (for example when controlling the operating point of a column) it may result useful to reintroduce the calculated values as starting point for the new calculus

Methods for solving staged columns (continuation)

Codes for variables

reboiler	1
total condenser	2
partial condenser	3

Codes for specifications

reflux ratio	1
product to feed ratio (molar fract.)	2
bottom to feed ratio (molar fract.)	3
component (molar fract.) in top product	4
component (molar fract.) in bottom product	5
component recovery in top product	6
component recovery in bottom product	7

Example

Column with 8 stages, 1 feed (stage 4), pressure reboiler 12.5 Bar, pressure top 12 Bar, stage efficiency 1.0, dH = 0.0
variables : reboiler and total condenser
specifications : component 2 fraction in top product and bottom product to to feed ratio

parameter	value	comment
csep	1	VLE column
stgnr	8	number of stages
init	0	automatic initialization
stgp[0]	12.5	pressure on stage 1
......		specify pressure for all stages
stgp[7]	12	pressure on stage 8
stgef[0]	1	efficiency on stage 1
......		specify efficiency for all stages
stgef[7]	1	efficiency on stage 8
stgdH[0]	0	heat added, removed on stage 1
......		specify heat added / removed for all stages
stgdH[7]	0	heat added, removed on stage 8
prod_h	1	product stream (Prode Properties stream 1)
btm_h	2	bottom stream (Prode Properties stream 2)
fnr	1	feeds number
fstr	3	feed stream (Prode Properties stream 3)
fpos	4	feed position
vnr	2	number of variables
vtype[0]	1	first variable, reboiler
vtype[1]	2	second variable, total condenser
ptype[0]	4	first specification, molar fraction in top product
piv[0]	2	first specification, second component (2) in the list
prv[0]	0.96	first specification, required fraction
ptype[1]	3	specification, bottom to feed ratio
piv[1]	0	not required
prv[1]	0.4	second specification, required value (bottom to feed ratio = 0.4)

Methods for solving reactors

simulation of reactors with standard procedures

```
int res = REACT(int streamIn, streamOut, int model, int NrReactions, double **Conv, double Pout, double dHeat)
```

Parameters :

streamIn	(int)	inlet stream
streamOut	(int)	outlet stream
model	(int)	model for reactor (see below)
NrReactions	(int)	number of reactions
Conv	(double**)	matrix (NrComponents, NrReactions) to specify reactions
Pout	(double)	output pressure
dHeat	(double)	heat added, removed

Codes for models

Gibbs	1
Equilibrium Reactor	2

(**) additional models available from Prode

Methods for solving fluid flow problems

simulation of single phase, two-phases, multiphase flow on circular pipes

```
int res = PIPE(int stream, int model, double diam, double rough, double length, double dHeight, double dHeat)
```

Parameters :

stream (int)	inlet stream
model (int)	model for fluid flow and phase equilibria (see the codes below)
diam (double)	pipe internal diameter
rough (double)	parameter defining relative pipe roughness
length (double)	lenght of this segment
dHeight (double)	height difference (inlet, outlet)
dHeat (double)	heat added, removed

codes for models

Beggs & Brill	1

(**) additional models available from Prode

Methods for Hydrates phase equilibria

methods for calculating hydrate formation pressure (or temperature)

```
double p = HPFORM(int stream, double t, int method)

double t = HTFORM(int stream, double p, int method)
```

Parameters :

stream	(int)	inlet stream
t, p	(double)	operating temperature (or operating pressure)
method	(int)	1 = include SI , SII , SH
		2 = SI
		3 = SII

Methods for solving Polytropic operations

Polytropic stage (compression and expansion) rigorous models for compressors and expanders including phase equilibria

double val = PSPF(int stream, double pout, int model, double param)

Parameters :

stream (int)	inlet stream
pout (double)	outlet pressure
model (int)	model, see below codes 1-4
param (double)	for model 1 and 3 specified polytropic efficiency (range 0-1)
	for model 2 and 4 (measured) outlet temperature

the procedure can model compression and expansion units such as centrifugal compressors, expansion turbines etc. including phase equilibria

the procedure returns
-calculated temperature options 1,3
-calculated efficiency options 2,4

1	given initial condition, pout and polytropic efficiency calculates outlet condition, R.A. Huntington “Evaluation of Polytropic calculation Methods for Turbomachinery Performance”, method applicable to gas phase only
2	given initial condition, pout and tout calculates polytropic efficiency, R.A. Huntington “Evaluation of Polytropic calculation Methods for Turbomachinery Performance”, method applicable to gas phase only
3	given initial condition, pout and polytropic efficiency calculates outlet condition, R.Paron “Polytropic solution with phase equilibria” method applicable to gas and mixed (gas + liquid) phases
4	given initial condition, pout and tout calculates polytropic efficiency, R.Paron “Polytropic solution with phase equilibria” method applicable to gas and mixed (gas + liquid) phases

(**) additional models available from Prode

Methods to design / rate orifices and relief valves

This unit models a relief valve (vapor and liquid phases) at specified operating conditions and returns the calculated area

double area = ISPF(int stream, double pout, int model, double *param)

Parameters :

stream (int)	inlet stream
pout (double)	outlet pressure
model (int)	model, see below codes 1-4
param(double)	correction parameter, see below the range of recommended values
models available (**)	
1	HEM Homogeneous Equilibrium (Solution of Mass Flux integral)
2	HNE Homogeneous Non-equilibrium (HEM with Boling Delay and Gas-Liquid Slip Contributes)
3	HNE-DS, Homogeneous Non-equilibrium
4	NHNE Non-homogeneous Non-equilibrium

recommended range of values for correction parameter

HEM	not required
HNE	0.7-0.8 for safety valves
HNE-DS	see the paper
NHNE	0.7-0.8 for safety valves

(**) additional models available from Prode

Methods for calculating equilibrium lines in phase diagrams

Prode Properties includes methods for calculating different types of phase diagrams
vapor-liquid
vapor-liquid-liquid
vapor-liquid-solid (**)
(**) feature available in custom versions

typical application
• define the stream, set the required phase equilibria (vapor-liquid, vapor-liquid-liquid, vapor-liquid-solid)
• call PELnr() to calculate the phase diagram and obtain the number of lines available
• on each line call PELP(), PELT(), PELine() to obtain the data for the different lines
• if required call PFLine() to calculate a line with specified phase fraction ad state

integer lnr = PELnr(integer stream)
sname xpelnr
Given a stream calculates the phase diagram and returns the number of equilibrium lines available

integer lnr = PELT(integer stream, integer line)
sname xpelt
Given a stream and the line, returns the line type (see below)
1 = bubble line
2 = dew line
3 = three phase line

integer lnr = PELP(integer stream, integer line)
sname xpelp
Given a stream and the line, returns the line property (see below)
1 = vapor-liquid
2 = vapor-liquid-liquid
3 = vapor-solid
4 = liquid-solid

integer nrpt =PELine(integer stream, integer line, double *P, double *T, int maxpt)
sname xpel
Given a stream, the line and two arrays (0 -maxpt elements) the procedure returns nrpt < maxpt equilibrium points in specified line

integer nrpt =PVLine(integer stream, integer line, double *P, double *T, double *H, double *S,double *V,int maxpt)
sname xpeel
Given a stream, the line and five arrays (0 -maxpt elements) the procedure returns nrpt < maxpt equilibrium points in specified line, in additions to t,p values this method returns enthalpy, entropy and volume values calculated at equilibrium points

integer nrpt =PFLine(integer stream,int line, double pf, double *P, double *T, int maxpt)
sname xpepfl
Given a stream, the line, a specified phase fraction and two arrays (0-maxpt elements) the procedure returns nrpt < maxpt equilibrium points in specified phase fraction line

Methods for direct access to properties (F,H,S,V) and derivatives (T,P,W)

Prode Properties includes methods for fast calculations of thermodynamic properties, you can define up to 5 independent processes with method DPinit(), these processes run independently permitting fast executions.

Application example :

```
Process = 1;                    // range 1-5
Stream=5;                       // make sure stream 5 has been defined before to call Dpinit()
DPinit(process,stream);
StrHv(process,0,t ,p,X,&HL);
StrHv(process,1,t ,p,Y,&HV);
```

integer res = DPinit(integer process,integer stream)
sname xspi
Given a process (code 1-5) and a stream the method loads all data

integer res = StrFv(integer process,integer state,double t ,double p, double *w,double *fg)
sname xsfv
Given a predefined stream the required state and operating conditions returns the vector of fugacities(Pa)

integer res = StrFvd(integer process,integer state,double t ,double p, double *w,double *fg, double *dfgt, double *dfgp, double **dfgw)
sname xsfvd
Given a predefined stream the required state and operating conditions returns the vector of fugacities (Pa) and related derivatives vs. temperature (K), pressure (Pa), composition (note : derivatives vs. composition as matrix [n][m])

integer res = StrFvdv(integer process,integer state,double t ,double p, double *w,double *fg, double *dfgt, double *dfgp, double *dfgw)
sname xsfvdv
Given a predefined stream the required state and operating conditions returns the vector of fugacities (Pa) and related derivatives vs. temperature (K), pressure (Pa), composition (note : derivatives vs. composition as vector [n*m])

integer res = StrHv(integer process, integer state,double t ,double p, double *w,double *H)
sname xshv
Given a predefined stream the required state and operating conditions returns the molar enthalpy (Kj/ Kmol)

integer res = StrHvd(integer process,integer state,double t ,double p, double *w,double *H, double *dHt, double *dHp, double *dHw)
sname xshvd
Given a predefined stream the required state and operating conditions returns the molar enthalpy (Kj/ Kmol) and related derivatives vs. temperature, pressure, composition

integer res = StrSv(integer process,integer state,double t ,double p, double *w,double *S)
sname xssv
Given a predefined stream the required state and operating conditions returns the molar entropy (Kj/ Kmol-K)

integer res = StrSvd(integer process,integer state,double t ,double p, double *w,double *S, double *dSt, double *dSp, double *dSw)
sname xssvd
Given a predefined stream the required state and operating conditions returns the molar entropy (Kj/ Kmol-K) and related derivatives vs. temperature, pressure, composition

integer res = StrVv(integer process,integer state,double t ,double p, double *w,double *V)
sname xsvv
Given a predefined stream, the required state and operating conditions returns the molar volume (M3/Kmol)

integer res = StrVvd(integer process,integer state,double t ,double p, double *w,double *V, double *dVt, double *dVp, double *dVw)
sname xsvvd
Given a predefined stream the required state and operating conditions returns the molar volume (M3/ Kmol) and related derivatives vs. temperature, pressure, composition

Methods for stream’ s data access

Extended methods to obtain properties

These methods are equivalent to standard methods but they add the operating conditions at which the required property must be evaluated. This may result useful in many cases, for example when utilizing Prode Properties methods as macros from Excel cells.

double mw = EStrGMw(integer stream, double t, double p)

sname xstpmw

given the stream, pressure and temperature performs an isothermal flash and returns the molecular weight for gas phase

double mw = EStrLMw(integer stream, double t, double p)

sname xstplmw

given the stream, pressure and temperature performs an isothermal flash and returns the molecular weight for liquid phase

double lf = EStrLf(integer stream, double t, double p)

sname xstplf

given the stream, pressure and temperature performs an isothermal flash and returns liquid fraction (molar basis) in stream

double pf = EStrPf(integer stream, integer state, double t, double p)

sname xstppf

given a stream , state (gas, liquid, solid) pressure and temperature performs an isothermal flash and returns the phase fraction (molar basis) in specified state

double zv = EStrZv(integer stream, double t, double p)

sname xstpzv

given the stream, pressure and temperature performs an isothermal flash and returns the relevant compressibility factor (gas phase)

double h = EStrH(integer stream, double t, double p)

sname xstph

given the stream, pressure and temperature performs an isothermal flash and returns the enthalpy (gas + liquid phase)

double v = EStrV(integer stream, double t, double p)

sname xstpv

given a stream, pressure and temperature performs an isothermal flash and returns the specific volume as sum of specific volumes of all phases

double cp = EStrGCp(integer stream, double t, double p)

sname xstpgcp

given the stream, pressure and temperature performs an isothermal flash and returns the specific heat capacity (constant pressure, gas phase)

double cv = EStrGCv(integer stream, double t, double p)

sname xstpgcv

given the stream, pressure and temperature performs an isothermal flash and returns the specific heat capacity (constant volume, gas phase)

double cp = EStrLCp(integer stream, double t, double p)

sname xstplcp

given the stream, pressure and temperature performs an isothermal flash and returns the specific heat capacity (constant pressure, liquid phase)

double cv = EStrLCv(integer stream, double t, double p)

sname xstplcv

given the stream, pressure and temperature performs an isothermal flash and returns the specific heat capacity (constant volume, liquid phase)

double c = EStrGIC(integer stream, double t, double p)
sname xstpgic
given the stream, pressure and temperature performs an isothermal flash and returns the isothermal compressibility in gas phase

double c = EStrLIC(integer stream, double t, double p)
sname xstplic
given the stream, pressure and temperature performs an isothermal flash and returns the the isothermal compressibility in liquid phase

double ss = StrMSS(integer stream, double t, double p)
sname xstpmss
given the stream pressure and temperature performs an isothermal flash and returns returns the speed of sound (gas, liquid) as calculated with HEM model for mixed phases

double ss = EStrGSS(integer stream, double t, double p)
sname xstpgss
given the stream, pressure and temperature performs an isothermal flash and returns the speed of sound in gas phase

double ss = EStrLSS(integer stream, double t, double p)
sname xstplss
given the stream, pressure and temperature performs an isothermal flash and returns the speed of sound in liquid phase

double jt = EStrGJT(integer stream, double t, double p)
sname xstpgjt
given the stream, pressure and temperature performs an isothermal flash and returns the Joule Thomson coefficient for gas phase

double jt = EStrLJT(integer stream, double t, double p)
sname xstpljt
given the stream, pressure and temperature performs an isothermal flash and returns the Joule Thomson coefficient for liquid phase

double ic = EStrGIC(integer stream double t, double p)
sname xstpgic
given the stream, pressure and temperature performs an isothermal flash and returns the isothermal compressibility coefficient (1 / V) * dV / dP in gas phase

double ic = EStrLIC(integer stream double t, double p)
sname xstplic
given the stream, pressure and temperature performs an isothermal flash and returns the isothermal compressibility coefficient (1 / V) * dV / dP in liquid phase

double v = EStrGVE(integer stream double t, double p)
sname xstpgve
given the stream, pressure and temperature performs an isothermal flash and returns the volumetric expansivity coefficient (1 / V) * dV / dT in gas phase

double v = EStrLVE(integer stream double t, double p)
sname xstplve
given the stream, pressure and temperature performs an isothermal flash and returns the volumetric expansivity coefficient (1 / V) * dV / dT in liquid phase

double hc = EStrHC(integer stream, double t, double p)
sname xstphc
given the stream, pressure and temperature performs an isothermal flash and returns the net heat of combustion (gas phase).

double fl = EStrFML(integer stream, double t, double p)
sname xstpfml
given the stream, pressure and temperature performs an isothermal flash and returns the flammability lean limit (gas phase).

double fl = EStrFMH(integer stream, double t, double p)
sname xstpfmh
given the stream, pressure and temperature performs an isothermal flash and returns the flammability rich limit (gas phase).

double s = EStrS(integer stream, double t, double p)
sname xstps
given the stream, pressure and temperature performs an isothermal flash and returns the relative entropy (gas + liquid phase)

double d = EStrLD(integer stream, double t, double p)
sname xstpld
given the stream, pressure and temperature performs an isothermal flash and returns the calculated liquid density (at operating conditions).

double d = EStrGD(integer stream, double t, double p)
sname xstpgd
given the stream, pressure and temperature performs an isothermal flash and returns the calculated gas density (at operating conditions).

double tc = EStrLC(integer stream, double t, double p)
sname xstplcl
given the stream, pressure and temperature performs an isothermal flash and returns the calculated liquid thermal conductivity (at operating conditions).

double tc = EStrGC(integer stream, double t, double p)
sname xstpgc
given the stream, pressure and temperature performs an isothermal flash and returns the calculated gas thermal conductivity (at operating conditions).

double v = EStrLV(integer stream, double t, double p)
sname xstplv
given the stream, pressure and temperature performs an isothermal flash and returns the r calculated liquid viscosity (at operating conditions).

double v = EStrGV(stream, double t, double p)
sname xstpgv
given the stream, pressure and temperature performs an isothermal flash and returns the calculated gas viscosity (at operating conditions).

double st = EStrST(integer stream, double t, double p)
sname xstpst
given the stream, pressure and temperature performs an isothermal flash and returns the calculated surface tension (at operating conditions).

Methods for chemical’s file access

Prode Properties includes a set of functions for accessing data in chemical’s file. Components are referenced via a component code which is an integer with value in the range 1 to getFCNR()

Integer nr = getFCNr()
sname xfcnr
returns the number of components in Chemical’s File

int str = MCompF(integer code, char *s, integer slm)
given the component code fills string s with the relevant component formula (eventually truncated to slm maximum lenght) , this is the Microsoft specific method

char *str = CompF(integer code)
sname xcf
given the component code returns the relevant component formula (eventually truncated to string maximum length) , this is the ANSI C compatible method

int str = MCompN(integer code, char *s, integer slm)
given the component code fills string s with the relevant component name (eventually truncated to slm maximum length) , this is the Microsoft specific method

char *str = CompN(integer code)
sname xcn
given the component code returns the relevant component name (eventually truncated to string maximum length) , this is the ANSI C compatible method

int id = CompID(integer code)
sname xcid
given the component code returns component’s ID (it’s the CAS number)

int cc = CompCID(integer id)
sname xidc
given the component ID returns the component’s code

double mw = CompMw(integer code)
sname xcmw
given the component code returns the relevant molecular weight

double tc = CompTc(integer code)
sname xctc
given the component code returns the relevant critical temperature

double ac = CompAc(integer code)
sname xcac
given the component code returns the relevant acentric factor

double vc = CompVc(integer code)
sname xcvc
given the component code returns the relevant critical volume

double pc = CompPc(integer code)
sname xcpc
given the component code returns the relevant critical pressure

double dm = CompDm(integer code)
sname xcdm
given the component code returns the dipole moment

double rg = CompRg(integer code)
sname xcrg
given the component code returns the radius of gyration

double sol = CompSol(integer code)
sname xcsol
given the component code returns the solubility parameter

double hf = CompHf(integer code)
sname xchf
given the component code returns the std. enthalpy of formation

double gf = CompGf(integer code)
sname xcgf
given the component code returns the Gibbs energy of formation

double sf = CompSf(integer code)
sname xcsf
given the component code returns the enthalpy of fusion

double nb = CompNb(integer code)
sname xcnb
given the component code returns the normal boiling point

double mp = CompMp(integer code)
sname xcmp
given the component code returns the melting point

double p = CompVP(integer code, double t)
sname xcvp
given the component code and a temperature, returns the calculated saturation pressure (calculated via Chemical's file temperature dependent correlation)

double h = CompHG(integer code, double t0, double t1)
sname xchg
given the component code , initial and final temperatures for integration, returns the calculated ideal gas enthalpy (calculated via Chemical's file temperature dependent correlation)

double s = CompSG(integer code, double t0, double t1)
sname xcsg
given the component code , initial and final temperatures for integration, returns the calculated ideal gas entropy (calculated via Chemical's file temperature dependent correlation)

double h = CompHL(integer code, double t0, double t1)
sname xchl
given the component code , initial and final temperatures for integration, returns the calculated ideal liquid enthalpy (calculated via Chemical's file temperature dependent correlation)

double s = CompSL(integer code, double t0, double t1)
sname xcsl
given the component code , initial and final temperatures for integration, returns the calculated ideal liquid entropy (calculated via Chemical's file temperature dependent correlation)

double h = CompHS(integer code, double t0, double t1)
sname xchs
given the component code , initial and final temperatures for integration, returns the calculated ideal solid enthalpy (calculated via Chemical's file temperature dependent correlation)

double s = CompSS(integer code, double t0, double t1)
sname xcss
given the component code , initial and final temperatures for integration, returns the calculated ideal solid entropy (calculated via Chemical’s file temperature dependent correlation)

double h = CompHV(integer code, double t)
sname xchv
given the component code and a temperature, returns the calculated latent heat (calculated via Chemical’s file temperature dependent correlation)

double v = CompLV(integer code, double t)
sname xclv
given the component code and a temperature, returns the calculated liquid viscosity (calculated via Chemical’s file temperature dependent correlation)

double v = CompGV(integer code, double t)
sname xcgv
given the component code and a temperature, returns the calculated gas viscosity (calculated via Chemical’s file temperature dependent correlation)

double d = CompLD(integer code, double t)
sname xcld
given the component code and a temperature, returns the calculated liquid density (calculated via Chemical’s file temperature dependent correlation)

double tc = CompLC(integer code, double t)
sname xclc
given the component code and a temperature, returns the calculated liquid (thermal) conductivity (calculated via Chemical’s file temperature dependent correlation)

double tc = CompGC(integer code, double t)
sname xcgc
given the component code and a temperature, returns the calculated gas (thermal) conductivity (calculated via Chemical’s file temperature dependent correlation)

double st = CompST (integer code, double t)
sname xcst
given the component code and a temperature, returns the calculated surface tension (calculated via Chemical’s file temperature dependent correlation)

double d = CompSD(integer code, double t)
sname xcsd
given the component code and a temperature, returns the calculated solid density (calculated via Chemical’s file temperature dependent correlation)

double tc = CompSC(integer code, double t)
sname xcsc
given the component code and a temperature, returns the calculated solid (thermal) conductivity (calculated via Chemical’s file temperature dependent correlation)

Methods to set / access options / settings

To set / access the different options / settings the library includes specific methods,
getKO(), putKO() these methods accept and return a integer (32 bit) which contains all the settings (each bit in the integer represents a different option / setting)
getKS(), putKS() allow to access or define each option

int value = getKO(integer stream)
sname xsk
given a stream returns a code (integer) with the options

integer res = putKO (integer stream, integer value)
sname xsetsk
given a stream define the options

int value = getKS(integer stream, integer option)
sname xsks
given a stream and option (see below the codes) returns a boolean (0-1) with stored value

integer res = putKS (integer stream, integer option, integer value)
sname xsetsks
given a stream and option define the option.

Table of codes to specify the different options

reference : methods getKO(), setKO() ...

Bit	Decimal value	Option
1	1	set multiphase vapor + liquid
2	2	set multiphase vapor + liquid + solid
3	4	set multiphase vapor + liquid + solid + hydrate
4	8	reduce the number of trial phases (in multiphase)
5	16	use iso compressibility coeff. to detect single phase state
6	32	evaluate stability of each phase in equilibrium
7	64	end specified phase fraction lines when crossing phase boundary lines
8	128	include all hydrate structures (also those not normally generated by formers)

to set one or more options call setOM() passing as value a integer with the sum (decimal values) of all required options.

Table of codes to specify the different states

reference : methods setMP() , PfTF() , PfTF() , StrFv(), StrFvd() ...

Code	State
0	Vapor phase
1	Liquid phase
2	Solid phase
3	Hydrate phase

Access to specific values

double p = getPatm()
sname xpatm
returns the internal reference (user defined) for atmosferic pressure quantity.

Table of codes to specify the different models

reference : methods setMP(), getMP() ...

Some models may not be available and/or the numerical codes may change in different versions, contact Prode for details

Code	Description	mixing rules	Model
1	Regular		Regular
10	Wilson		Wilson
11	NRTL		NRTL
12	UNIQUAC		UNIQUAC
30	Soave-Redlich_Kwong	VDW	SRK(VDW)
31	Soave-Redlich_Kwong ext.	VDW	SRKX(VDW)
40	Soave-Redlich_Kwong ext. + NRTL	P-HV	SRKX-NRTL(P-HV)
41	Soave-Redlich_Kwong ext. + NRTL	P-LCVM	SRKX-NRTL(P-LCVM)
50	Peng Robinson std.	VDW	PR(VDW)
51	Peng Robinson ext.	VDW	PRX(VDW)
55	Peng Robinson ext. + Wilson	WS	PRX-Wilson(WS)
56	Peng Robinson ext. + UNIQUAC	WS	PRX-UNIQUAC(WS)
57	Peng Robinson ext. + NRTL	WS	PRX-NRTL(WS)
60	Peng Robinson ext. + NRTL	P-HV	PRX-NRTL(P-HV)
61	Peng Robinson ext. + Wilson	P-HV	PRX-Wilson(P-HV)
62	Peng Robinson ext. + UNIQUAC	P-HV	PRX-UNIQUAC(P-HV)
65	Peng Robinson ext. + Wilson	MHV2	PRX-Wilson(MHV2)
66	Peng Robinson ext. + UNIQUAC	MHV2	PRX-UNIQUAC(MHV2)
67	Peng Robinson ext. + NRTL	MHV2	PRX-NRTL(MHV2)
70	Peng Robinson ext. + NRTL	P-LCVM	PRX-NRTL(P-LCVM)
71	Peng Robinson ext. + Wilson	P-LCVM	PRX-Wilson(P-LCVM)
72	Peng Robinson ext. + UNIQUAC	P-LCVM	PRX-UNIQUAC(P-LCVM)
73	Peng Robinson ext. + UNIFAC	P-LCVM	PRX-UNIFAC(WS)
80	Benedict-Webb-Rubin		BWR
81	Benedict-Webb-Rubin-Starling		BWRS
90	Lee Kesler		LK
91	Lee Kesler Ploecker		LKP
100	P-SAFT		PSAFT
111	Peng Robinson ext. inc. association (CPA)	VDW	PRX-CPA(VDW)
115	Peng Robinson ext. inc. association (CPA) + NRTL	P-HV	PRXCPA-NRTL(P-HV)
116	Peng Robinson ext. inc. association (CPA) + NRTL	P-LCVM	PRXCPA-NRTL(P-LCVM)
117	Peng Robinson ext. inc. association (CPA) + NRTL	MHV2	PRXCPA-NRTL(mod.P-MHV2)
118	Peng Robinson ext. inc. association (CPA) + NRTL	WS	PRXCPA-NRTL(P-WS)
130	UNIFAC		UNIFAC
150	Solid Pure (derived from) PRX-NRTL(P-HV)		SPRX-NRTL(P-HV)
151	Solid Pure (derived from) PRXCPA-NRTL(P-HV)		SPRXCPA-NRTL(P-HV)
153	Solid Solution (derived from) PRX-NRTL(P-HV)		SSPRX-NRTL(P-HV)
170	Hydrate (derived from) PRXCPA-NRTL(P-HV)		HPRXCPA-NRTL(P-HV)
171	Hydrate (derived from) PRX-NRTL(P-HV)		HPRX-NRTL(P-HV)
180	Wax		Wax
185	Asphaltene		Asphaltene
200	Pitzer (Electrolyte)		PITZER
205	Peng Robinson ext. Ass. / MSA / NRTL		PRXCPA-E-NRTL(P-HV)
210	P-SAFT-(MSA) Electrolyte		PSAFT-E
300	Steam tables IAPWS 1995		IAPWS 95
311	GERG 2008 / AGA 2017		GERG 2008
312	ISO 18453 (GERG)		ISO 18453
315	ISO 20765 (AGA 8)		ISO 20765

Auxiliary methods

Thermodynamic models

To define or retrieve the thermodynamic models associated with each property (Fg, H, S,V..) of a stream the library includes setMP(), getMP()

integer res = setMP(integer stream, integer mp, integer state, integer model)
sname xsetsm
given a stream, property (Fg,H,S..) model and state (Vapor,Liquid,Solid,Hydrate) this method sets the specified model for that property and returns TRUE in case of success, otherwise returns FALSE

integer m = getMP(integer stream, integer mp, integer state)
sname xsm
given a stream, related property (Fg,H,S..) and state (Vapor,Liquid,Solid,Hydrate) this method returns the specified model for that property and state

Table of codes to specify the different properties in setMP() and getMP()

1	Fugacity
2	Enthalpy
3	Entropy
4	Volume
5	Viscosity

Base values for enthalpy / entropy

Prode Properties allows the user to define the base values (the temperature and initial value from which to start integration) for entropy and enthalpy from Properties Editor, in setting's page, these values are stored in archive and restored when program starts.
In addition it is possible to modify these value by code with the following methods,

integer res = setHB(integer mod, double t, double val)
sname xsethb
given a code to identify the procedure (see the table with codes), the temperature and initial value sets base value for enthalpy .

integer res = setSB(integer mod, double t, double val)
sname xsetsb
given a code to identify the procedure (see the table with codes), the temperature and initial value sets base value for entropy .

Table of codes to specify the different base values in setHB() and setSB()

1	initial values specified by user (values of t and val)
2	initial values are enthalpy of formation (or entropy of formation) and temperature 25 C

Stream names

In Prode Properties streams have several properties including a label (name) which could match (for example) the name of a line in your project, you can easily set / access these labels through a series of methods.

integer str = MStrN(integer stream, char *s, integer slm)
given a integer (that identifies a stream) method fills string s with the name of stream (eventually truncated to slm maximum lenght), this is the Microsoft specific method

char *str = StrN(integer stream)
sname xsn
given a integer (that identifies a stream) method returns as ANSI C type the string identifying that stream.

integer res = putN(integer stream, char *str)
sname xsetsn
given a integer (that identifies a stream) and a ANSI C string identifying that stream this method sets the label.

Methods to access Model's data

Prode Properties includes models for calculating properties as fugacities, enthalpies, entropies, volumes, viscosities etc. these methods allow to access the models available

integer nr = getMDnr()
sname xmdnr
returns the number of models available in the library

char *str = getMDN(int model)
sname xmdn
given the model position (in the range 1-number of models available) the method returns the name of model.

integer res = getMDP(int model, int prop, int state)
sname xmdp
given the model position (in the range 1-number of models available) the property and state returns TRUE if model can calculate the specified property, otherwise returns FALSE

integer code = getMDC(int model)
sname xmdc
given the model position (in the range 1-number of models available) returns the code of the model

Methods to control error's messages

The library includes functions to control the error messages

setErrFlag (integer state)
sname xseterr
given a Boolean (state) sets the error flag to TRUE or FALSE. The flag should be cleared (state = FALSE) before each sequence of calculations and tested (method getErrFlag()) after the calcs. If this is done, then a flag state of TRUE indicates that an error has occurred somewhere in the calculation sequence).

integer res = getErrFlag ()
sname xerr
a value of TRUE means that an error has been found, please note that PROPERTIES doesn't clear the error flag state, You should clear the error flag (via setErrFlag()) before each sequence of calc's.

integer str = MErrMsg(char *s, integer slm)
fills string s with the last error message generated (eventually truncated to slm maximum lenght), this is the Microsoft specific method

char *str = ErrMsg()
sname xerrmsg
Returns the last error message generated, this is the ANSI C compatible method

Methods for accessing Prode Editor

Prode Properties includes methods to open programmatically Properties Editor

integer res = edS(integer stream)
sname xeds
given a integer (that identifies a stream) method activates the Properties Editor on the specified stream

integer res = edSS()
sname xedss
this method activates the Properties Editor on first stream

Methods to load / save archives

Archives are files which contain a copy of all stream's, units of measurement, settings etc. stored in Prode Properties memory when the file was created.

When you load an archive all data will be restored, archives are useful to create copies of your work which would otherwise be lost when leaving the application, Prode Properties includes methods for operations on archives.

integer res = AOpen()
sname xaopen
open a file as archive (browse for file)

integer res = AFOpen(char *path)
sname xafopen
open the file specified in *path as archive

integer res = ASave()
sname xasave
save a file as archive (browse for file)

integer res = AFSave(char *path)
sname xafsave
save the file specified in *path as archive

Methods for accessing / defining the units of measurement

Prode Properties includes methods for accessing and defining the units of measurement, these methods utilize a numeric code for identifying the correspondent quantities, refer to the paragraph "Access via software to the units of measurement" for a list of these codes.

integer res = getUMC(integer UM)
sname xumc
given a integer (that identifies a quantity) method returns the selected UM for that quantity.

integer res = setUMC(integer UM, integer sel)
sname xsetumc
given two integers (the first identifies a quantity and the second the selection) method selects a UM for that quantity.

integer   res = getUMN(integer UM)
sname xumn
given a integer (that identifies a quantity) method returns the number of different units of measurement available for that quantity.

integer str = MgetUMS(integer UM, integer sel, char *s, integer slm)
given two integers (the first identifies a quantity and the second the selection) fills string s with selected UM (eventually truncated to slm maximum lenght), this is the Microsoft specific method

char *str = getUMS(integer UM, integer sel)
sname xums
given two integers (the first identifies a quantity and the second the selection) method returns as ANSI C type the string identifying the selected UM.

integer str = MgetSUMS(integer UM, char *s, integer slm)
given a integer UM for quantity fills string s with selected UM (eventually truncated to slm maximum lenght), this is the Microsoft specific method

char *str = getSUMS(integer UM)
sname xsums
given a integer UM for quantity this method returns as ANSI C type the string identifying the selected UM.

double res = UMCR(double value, integer UM, integer SEL)
sname xumcr
given a value, the code for quantity and selection converts to reference and returns the result

double res = UMCS(double value, integer UM, integer SEL)
sname xumcs
given a value, the code for quantity and selection converts from reference and returns the result

integer res = UMAU(double a, double b, char *name, integer UM)
sname xumau
given the code for a quantity, the parameters a, b required for conversion and the name adds a new (user defined, temporary) unit.

integer res = UMRAU(integer UM)
sname xumrau
given the code for a quantity removes all additional (temporary) units

Units of measurement

Prode Properties allows to define via software the units of measurement, see paragraph “Methods for accessing / defining the units of measurement”, in Prode Properties to reference a unit must use a numeric code

QUANTITY	UNIT DEF	NUMERIC CODE	DEFAULT UNIT
Pressure (abs)	CONV_P	15	“Pa.a”
Pressure (rel)	CONV_DP	16	“Pa”
Temperature (abs)	CONV_T	17	“K”
Temperature(rel)	CONV_DT	18	“K”
Calorific Value (weight)	CONV_HM	19	“Kj/Kg”
Calorific Value (molar)	CONV_HMM	20	“Kj/Kmol”
Power	CONV_HS	21	“KW”
Entropy (Streams)	CONV_SS	22	“KJ/(K*s)”
Heat Capacity (weight)	CONV_CP	23	“kJ/(kg*K)”
Heat Capacity (molar)	CONV_CPM	24	“kJ/(kmol*K)”
Flow (mass basis)	CONV_W	25	“Kg/s”
Flow (gas, mass basis)	CONV_WG	26	“Kg/s”
Density (weight)	CONV_D	27	“Kg/m3”
Density (molar)	CONV_DM	28	“Kmol/m3”
Specific Volume (weight)	CONV_SV	29	“m3/Kg”
Specific Volume (molar)	CONV_SVM	30	“m3/Kmol”
Thermal Conductivity	CONV_TC	31	“W/(m*K)”
Viscosity (dynamic)	CONV_V	32	“Pa*s”
Surface Tension	CONV_ST	33	“N/m”
Lenght	CONV_L	34	“m”
Area	CONV_A	35	“m2”
Volume	CONV_VOL	36	“m3”
Mass	CONV_M	37	“Kg”
Velocity	CONV_VL	38	“m/s”
Acceleration	CONV_ACC	39	“m/s2”
Force	CONV_FOR	40	“N”
Time	CONV_TM	41	“s”
Heat Flux	CONV_HF	42	“KW/m2”
Thermal Resistance	CONV_TR	43	“K*m2/KW”
Heat Transfer Coefficent	CONV_HTC	44	”KW/(m2*C)”
Flow (volume basis)	CONV_VW	45	“m3/s”
Viscosity (kinematic)	CONV_VK	46	“m2/s”
Energy	CONV_EN	47	“KJ”
Dipole moment	CONV_EDM	48	“c-m”
Solubility parameter	CONV_SP	49	“(J/m3)^1/2”
Flow Coefficient	CONV_CV	50	“Cv”
Compressibility coefficient	CONV_CC	51	“1/Pa”
Joule Thomson coefficient	CONV_JTC	52	“K/Pa”
Flow (molar basis)	CONV_WM	53	“Kmol/s”
Volume expansivity	CONV_VE	54	“1/K”

Error messages

Prode Properties may generate one or more error messages, herebelow a short list with possible causes

Memory allocation error
• limit in resources allocation (close applications, release memory and restart)

Corrupted file, error reading data file
• the library cannot access a file, this may depend from the file not being in the proper directory or being corrupted, reinstall the software

Internal error
• this error may depend from several different causes, for example a wrong parameter in a function (i.e. an attempt to pass a value out of permitted range).

too many local variables
• too many variables
• a limit in resources allocation

calc. on undefined stream data
• an undefined stream found while executing calc's (edit and define the stream)

undefined stream' s operating conditions
• pressure, temperature or flow are undefined (edit and define the stream)

error calling thermo calc. procedure
• wrong input value (calcs cannot converge) or calcs outside temperature range (check chemical's file for limits in temperature correlation's).

cannot converge calc' s loop
• a wrong convergence condition has been specified, for example a parameter outside the correct range etc.

T, P values outside H, S range calcs
• a wrong condition has been specified, for example t, p outside range limits

too many comp' s in a stream
• when two or more streams are mixed the total nr. of components may exceed the maximum
• inconsistencies in stream's data

error accessing component' s data archive
• unavailable data (a unspecified component) or calc's outside temperature range.

Stack error (no memory), reload procedure
• a limit in resources allocation (see above)

Method not available in this version
Attempt to define a method not available in that version, edit the stream and define a new method

A stream with Steam Tables model must have only 1 component
• steam tables model requires one component only (water)

Calculation basis

The user can specify which method to use selecting the models.
Please refer to the paragraph "reference literature" and "Models" for additional information about the methods.

Fugacity	calculated according selected model
Enthalpy	calculated according selected model
Entropy	calculated according selected model
Volume	calculated according selected model

Viscosity
gas low pressure mixing rule according Wilke (1950) , operating conditions correction according Stiel and Thodos (1964)
liquid logarithmic average mixing rule, pressure correction according Lucas (1981)

Thermal conductivity
gas low pressure mixing rule according Mason and Saxena (1958), operating conditions correction according Stiel and Thodos (1964)
liquid mixing rule according Li (1976)

Surface tension
mixing rule according MacLeod-Sugden

Heat of combustion
weight average mixing rule according ISO std. (database contains values in Kj/Kg)

Flammability limits
mixing rule according Le Chatelier as discussed by Coward & Jones (1952)

Enthalpy, Entropy calc's
the user can specify different initial conditions for enthalpy and entropy, see "Prode Editor : Config Page" for additional details.

Temperature, pressure ranges

Temperature range	1 K - 5000 K
Pressure range	1 Pa - 1000 Bar

Chemical's File format

This section discusses the file format adopted by Prode Properties to store chemical’s data, the program stores for each component a large number of data as shown in following list, data is stored in a binary compressed format.

Prode Properties allows to select different correlations to define each temperature dependent property, all major standards including DIPPR are supported
Note that data dependent correlation's have a range of temperature for application, Prode Properties includes tests for this range (as defined by high and low limits in chemicals file) and, when required, attempts to estimate the values outside this range, in some cases this may produce inconsistent results.

Prode Properties base version adopts the following format

Formula string 12 chars max
Name (1) (main list) string 40 chars max
Name (2) (user defined list) string 40 chars max
Name (3) (user defined list) string 40 chars max
Identification number (CAS as default)
Molecular weight
Critical temperature
Critical pressure
Critical volume
Acentric factor
Dipole Moment
Radius of Gyration
Solubility parameter
Standard enthalpy of formation (298 K)
Gibbs free energy of formation (298 K, 1 atm)
Enthalpy of fusion
Normal boiling point
Melting point
Flammability lean limit % (range 0-100)
Flammability rich limit % (range 0-100)
Autoignition temperature
Net heat of combustion
Flash Point

Gas heat capacity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Vapor viscosity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Vapor thermal conductivity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Heat of vaporization correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Liquid vapor pressure correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Surface tension
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Liquid density correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Liquid viscosity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Liquid thermal conductivity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Liquid heat capacity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Solid vapor pressure correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Solid density correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Solid thermal conductivity correlation
type of equation
unit for property
unit for temperature
low temperature limit
high temperature limit
A-E (5 parameters)

Solid heat capacity correlation
type of equation
unit for property
unit for temperature
low temperature limit
igh temperature limit
A-E (5 parameters)

Sources of data

Data in chemical data file come from several sources including :

• “Dechema Chemistry Data ser.” text books

• “DIPPR data collection” text books

• “Technical Data Book, Petroleum Refining”

Due to the large differences in critical and transport properties found in different sources, DIPPR (AICHE Design Institute for Physical PRoperty Data) reference has been selected as a default.

Component’s identification

Components are identified by name (from DIPPR list) , chemical formula and Identification number.

Regression procedures and results

Coefficients in correlations have been calculated with a custom program that uses a modified version of Levenberg-Marquardt algorithm , reported errors (at each fitting point) are usually lower than 1 % of input values for the most complex correlations (i.e. vapor pressure),), however in some cases they may be higher.

Consistency tests

When relations exist between thermodynamic properties (i.e. acentric factor and critical pressure and temperature, vapor pressure and heat of vaporization etc.) a consistency test has been performed.

Comparing Prode Properties results against those of different process simulators

When comparing data from different tools one must verify that

• the different tools do use the same thermodynamic models

• properties in databanks have siimilar values

• lists and values of BIPs and other parametres which can influence results have similar values

Models

Prode Properties includes a complete set of thermodynamic models (some available in extended versions)

Liquid activities
Wilson
NTRL
UNIQUAC
Predictive
UNIFAC
Electrolytes
Pitzer
CPA-electrolyte
SAFT-electrolyte

Cubic EOS
Soave-Redlich-Kwong, Peng-Robinson with std. alpha function and VdW mixing rules,
Extended versions of SRK and PR including parameters calculated to fit experimental data (saturation pressures, densities, heat capacities etc.) and different mixing rules to combine equations of state with activity models
Std. and Modified versions of Huron Vidal (HV) rule
Std. and Modified versions of Linear Combination of Vidal and Michelsen (LCVM) rule
Std. and Modified versions of Michelsen-Huron-Vidal (MHV2) rule
Std. and Modified versions of Wong Sandler (WS) rule
Other models
Modified Benedict-Webb-Rubin
Benedict-Webb-Rubin-Starling
Lee-Kesler
Lee-Kesler-Plocker

Models based on associating fluid theory
Different versions of CPA Cubic Plus Association based on Soave Redlich Kwong and Peng Robinson models with VdW mixing rules and several others to combine equations of state with activity models
Std. and Modified versions of Huron Vidal (HV) rule
Std. and Modified versions of Linear Combination of Vidal and Michelsen (LCVM) rule
etc...
Different versions of SAFT (Perrturbed Chain Statistical Associating Fluid Theory)

Solids
SPM (Solid Pure Model) solid phase treated as single component
SSM (Solid Solution Model) solid phase treated as homogeneous solution
WAX solid phase treated as homogeneous solution (with specific parameters)
Asphaltene
Hydrates (based on Van der Waals and Plateeuw theory with a std. model and a complex model)

Standards (based on international standards)
GERG 2008 (ISO 20765)
AGA 2017 (2017 version with GERG 2008 formulations)
Steam tables (IAPWS 1995) Water / steam properties calculated according IAPWS 1995 formulation

All the models included in Prode Properties export derivatives of Fg, H, S, V vs. W, P, T

UNIFAC functional groups

The underlying idea in UNIFAC method is that a molecule can be considered as a collection of functional groups. The main advantage of this approach is that from a relatively small number of functional groups the properties of many different molecules can be predicted. The UNIFAC model is useful for estimating solution behavior in the absence of experimental data.

Prode Properties incorporates the UNIFAC Group Contribution revision 5 (January 1992, J.P.Baker), following the main groups and subgroups table :

Code	Main	Subgroup	Example
1	CH2	CH3	Hexane
2		CH2	n-Hexane
3		CH	2-Methylpropane
4		C	Neopentane
5	C=C	CH2=CH	1-Hexene
6		CH=CH	2-Hexene
7		CH2=C	2-Methyl-1-butene
8		CH=C	2-Methyl-2-butene
70		C=C	2,3-Dimethylbutene
9	ACH	ACH	Naphthaline
10		AC	Styrene
11	ACCH2	ACCH3	Toluene
12		ACCH2	EthylBenzene
13		ACCH	Cumene
14	OH	OH	n-Propanol
15	CH3OH	CH3OH	Methanol
16	H2O	H2O	Water
17	ACOH	ACOH	Phenol
18	CH2CO	CH3CO	Butanone
19		CH2CO	Pentanone-3
20	CHO	CHO	Propionic aldehyde
21	CCOO	CH3COO	Butyl acetate
22		CH2COO	Methyl propionate
23	HCOO	HCOO	Ethyl formate
24	CH2O	CH3O	Dimethyl ether
25		CH2O	Diethyl ether
26		CHO	Diisopropyl ether
27		THF	Tetrahydrofuran
28	CNH2	CH3NH2	Methylamine
29		CH2NH2	Ethyl amine
30		CHNH2	Isopropylamine
31	CNH	CH3NH	Dimethylamine
32		CH2NH	Diethyl amine
33		CHNH	Diisopropylamine
34	(C)3N	CH3N	Trimethylamine
35		CH2N	Triethylamine
36	ACNH2	ACNH2	Aniline
37	Pyridine	C5H5N	Pyridine
38		C5H4N	2-Methyl pyridine
39		C5H3N	2,3-Dimethylpyridine
40	CCN	CH3CN	Acetonitrile
41		CH2CN	Propionitrile
42	COOH	COOH	Acetic acid
43		HCOOH	Formic acid
44	CCl	CH2Cl	Butane-1-chloro
45		CHCl	Propane-2-chloro
46		CCl	2-Methylpropane-2-chloro
47	CCl2	CH2Cl2	Methane-dichloro
48		CHCl2	Ethane-1,1-dichloro
49		CCl2	Propane-2,2-dichloro
50	CCl3	CHCl3	Chloroform

51		CCl3	Ethane-1,1,1-trichloro
52	CCl4	CCl4	Methane-tetrachloro
53	ACCl	ACCl	Benzene-chloro
54	CNO2	CH3NO2	NitroMethane
55		CH2NO2	Propane-1-nitro
56		CHNO2	Propane-2-nitro
57	ACNO2	ACNO2	Benzene-nitro
58	CS2	CS2	Carbon Disulfide
59	CH3SH	CH3SH	Methanethiol
60		CH2SH	Ethanethiol
61	Furfural	Furfural	Furfural
62	DOH	DOH	1,2-Ethanediol
63	I	I	Iodoethane
64	Br	Br	Bromoethane
65	C-C	CH-C	Hexyne-1
66		C-C	Hexyne-2
67	DMSO	DMSO	Dimethylsulfoxide
68	ACRY	Acrylnitril	Acrylnitrile
69	ClCC	Cl-(C=C)	Ethene-trichloro
71	ACF	ACF	Hexafluorobenzene
72	DMF	DMF-1	N,N-Dimethylformamide
73		DMF-2	N,N-Diethylformamide
74	CF2	CF3	Perfluorohexane
75		CF2	
76		CF	Perfluoromethylcyclohexane
77	COO	COO	Methyl acrylate
78	SiH2	SiH3	Methylsilane
79		SiH2	Diethylsilane
80		SiH	Heptamethyltrisiloxane
81		Si	Heptamethyldisiloxane
82	SiO	SiH2O	1,3-Dimethyldisiloxane
83		SiHO	1,1,3,3-Tetramethyldisiloxane
Code	Main	Subgroup	Example
84		SiO	Octamethylcyclotetrasiloxane
85	NMP	NMP	N-methylpyrrolidone
86	CClF	CCl3F	Trichlorofluoromethane
87		CCl2F	Tetrachloro-1,2-difluoroethane
88		HCCl2F	Dichlorofluoromethane
89		HCClF	1-Chloro-1,2,2,2,-tetrafluoroethane
90		CClF2	1,2-Dichlorotetrafluoroethane
91		HCClF2	Chlorodifluoromethane
92		CClF3	Chlorotrifluoromethane
93		CCl2F2	Dichlorodifluoromethane
94	CON	CONH2	Acetamid
95		CONHCH3	N-Methylacetamid
96		CONHCH2	N-Ethylacetamid
97		CON(CH3)2	N,N-Dimethylacetamid
98		CONCH3CH2	N,N-methylethylacetamid
99		CON(CH2)2	N,N-Diethylacetamid
100	OCCOH	C2H5O2	2-Ethoxyethanol
101		C2H402	2-Ethoxy-1-propanol
102	CH2S	CH3S	Dimethylsulfide
103		CH2S	Diethylsulfide
104		CHS	Diisopropylsulfide
105	Morpholine	MORPH	Morpholine
106	Thiophene	C4H4S	Thiophene
107		C4H3S	2-Methylthiophene
108		C4H2S	2,3-Dimethylthiophene

Prode Properties, last page