ProdeProperties Test File

for Prode Properties ver. 1.2b

This file may be used to insure that the Mathcad prode.dll file is in the right directory and that the functions are working properly.Prode Properties may be tested independent of Mathcad using the Excel version in the Prode directory.Examples are provided to demonstrate how the functions may be used.The process examples (compressor, nozzles, etc.) were based on the Excel examples included in the Prode Properties installation.

Directions for using this worksheet

The default archive from Prode, def.ppp, will be used as the starting file.This worksheet will write changes to a new archive, test.ppp.This file will be placed in the same directory as def.ppp, but the user may also later archive to other directories.

Operations that set a variable will show a result of 1 if successful or 0 if not.Results that retrieve values will show the retrieved value, or 0 if no value is available. The exceptions to this convention will be noted.

Automatic calculation has been turned off so the new user may read these instructions before starting the computations.

Procedure:
Calculate the entire worksheet, ctrl-F9
Select dep.ppp archive when first window appears, and click Open
When second popup window, the Prode archive, appears, it should show stream 2.Click OK.
Scroll through the worksheet to check for errors.See "Errors" below.
If the Prode window pops up, it is allowing you to view a recently created stream.Click OK to close.

Errors:
Mathcad will show errors in red as usual.Typical errors might be caused by syntax in the argument list, or the function may not be found in the library delivered by Prode, ppp.lib.
If the result is zero, then the Prode function had an error or could not return a value.Frequently, this may be caused by the lack of a particular phase needed for an operation.For example, solid properties can't be returned when a stream has no solid phase, or when the temperature is above the melting point for a pure compound property.
Rarely, a ppp.dll error window may appear with "Error accessing component's data archive".This appears to be caused by a lack of data in the chem.dat file for that particular property.If this window appears, it must be closed to proceed with the computations.See also the mc_defErrMsg function for a way to prevent these windows from stopping the calculations.The program is set to prevent these error windows.

File open commands

```
mc_AFOpen("C:\ProgramData\prode\def.ppp") = 1
```

disabled

This command sets the path to the archive file and directory.The path shown is the default set during the Prode Properties installation.This command will affect all subsequent uses of Prode on this computer until the path is changed again by another AFOpen command.Therefore, this command should be used with caution.

```
mc_AOpen("dummy") = 1
```

browse for an archive in the default directory

Note: Some of the Prode statements have been disabled to eliminate the repeated appearance of the Prode window during recalculation.

The next two functions do not obey the normal Prode convention regarding the result returned if successful. A result of 0 means these were successful. They must be evaluated (followed by "=") in order for the operation to take place.

`mc_setErrFlag(0)=0` set to 0at start of calc's to clear flag

$wopt := 0$

`mc_defErrMsg(wopt)=0` wopt = 0 turns off the Window Dialog messages
wopt = 1 turns on the Window Dialog messages

Open Properties window to view edit streams

$stream := 2$

`mc_edS(stream)=0` edit the given stream in the current active archive using the Prode window

`mc_edSS("dummy")=0` open to the first stream (disabled for the test to reduce popups)

Chemical file operations

`mc_getFCNr("dummy")=1635` number of components in data file, should be 1635 or greater.If less, then you are using the free version of Prode and not all of the routines below will work.

$id := 7732185$ CAS number of water (use internet search to find values for compounds)

compcode is an integer from 1 to number of components in the data file

`compcode := mc_CompCID(id)=1631` given id=CAS#, return compcode from database

Note: The above statement shows that the functions may be used to define a variable in addition to merely showing a result.

`mc_CompF(compcode)="H2O"` givena component code, returns component formula string

`mc_CompN(compcode)="WATER"` component name

$\mathrm{mc\_CompID}(compcode) = 7.7322 \cdot 10^{6}$ CAS number of component, compare to id above

Note: The units are not returned by the Prode commands.Operations that show which units are being used are shown later.

`mc_CompMw(compcode)=18.0153` molecular weight

$T_c := \mathrm{mc\_CompTc}(compcode)\ \mathrm{K} = 647.096\ \mathrm{K}$ critical temperature

$T_c = 64709.6008\ \mathrm{K\ s\ T\ R}$ multiply the function by the current Prode units for the result to use the unit features of Mathcad

$\mathrm{mc\_CompPc}(compcode) = 2.2064 \cdot 10^{7}$ critical pressure

`mc_CompVc(compcode)=0.0031` critical volume

`mc_CompAc(compcode)=0.344` acentric factor

$\mathrm{mc\_CompDm}(compcode) = 6.1782 \cdot 10^{-30}$	dipole moment
$\mathrm{mc\_CompRg}(compcode) = 6.15 \cdot 10^{-11}$	radius of gyration
$\mathrm{mc\_CompSol}(compcode) = 1511.8849$	solubility parameter
$\mathrm{mc\_CompHf}(compcode) = -13422.8364$	heat of formation
$\mathrm{mc\_CompGf}(compcode) = -12698.6729$	Gibbs energy of formation
$\mathrm{mc\_CompSf}(compcode) = 333.4738$	enthalpy of fusion
$\mathrm{mc\_CompNb}(compcode) = 373.15$	normal boiling point
$\mathrm{mc\_CompMp}(compcode) = 273.15$	melting point temperature

The following provide non zero values only if the phase of interest is present at the temperature requested.

$tgl := 300$	temperature for gas/liquids (above freezing for water)
$ts := 260$	temperature for solids (below freezing)
$\mathrm{mc\_CompVP}(compcode, tgl) = 3548.3262$	saturation pressure at temp tgl
$\mathrm{mc\_CompHV}(compcode, tgl) = 2436.313$	heat of vaporization at tgl
$\mathrm{mc\_CompLV}(compcode, tgl) = 0.0009$	liquid viscosity at tgl
$\mathrm{mc\_CompGV}(compcode, tgl) = 9.9253 \cdot 10^{-6}$	gas viscosity at tgl
$\mathrm{mc\_CompLD}(compcode, tgl) = 995.4764$	liquid density at tgl
$\mathrm{mc\_CompSD}(compcode, ts) = 918.6313$	solid density at ts
$\mathrm{mc\_CompLC}(compcode, tgl) = 0.6162$	liquid thermal conductivity at tgl
$\mathrm{mc\_CompGC}(compcode, tgl) = 0.0188$	gas thermal conductivity at tgl
$\mathrm{mc\_CompSC}(compcode, ts) = 0$	solid thermal conductivity at ts (appears to be missing for water)
$\mathrm{mc\_CompST}(compcode, tgl) = 0.0718$	surface tension at tgl

integrated changes between two temperatures, t0 and t1 for pure components

$t0 := 280$ $t1 := 290$

$\mathrm{mc\_CompHG}(compcode, t0, t1) = 18.6114$	ideal gas enthalpy change
$\mathrm{mc\_CompSG}(compcode, t0, t1) = 0.0653$	ideal gas entropy change
$\mathrm{mc\_CompHL}(compcode, t0, t1) = 42.0177$	ideal liquid enthalpy change
$\mathrm{mc\_CompSL}(compcode, t0, t1) = 0.1474$	ideal liquid entropy change
$ts0 := 260$ $ts1 := 270$	lower the temperature range < freezing pt

mc_CompHS(*compcode*, *ts0*, *ts1*) = 20.5241 ideal solid enthalpy change

mc_CompSS(*compcode*, *ts0*, *ts1*) = 0.0774 ideal solid entropy change

Units commands

See "Units of Measurement" section in Prode manual for a list of the units and their numerical codes.

UM := 15 pressure is used for an example

n_press := mc_getUMN(*UM*) = 22 no. of units avail. for UM

mc_getUMC(*UM*) = 1 present units code for UM

mc_getSUMS(*UM*) = "Pa.a" present units string for UM

sel := 5 select unit 5

mc_getUMS(*UM*, *sel*) = "KPa.a" units string for (UM, sel)

list all of the units for pressure

*i_* := [1..*n_press*]

$P\_units_{i\_} :=$ mc_getUMS(*UM*, *i_*)

$$P\_units = \begin{bmatrix} \text{"Pa.a"} \\ \text{"Pa.g"} \\ \text{"mbar.a"} \\ \text{"mbar.g"} \\ \text{"KPa.a"} \\ \text{"KPa.g"} \\ \text{"bar.a"} \\ \text{"bar.g"} \\ \text{"MPa.a"} \\ \text{"MPa.g"} \\ \text{"kgf/cmq.a"} \\ \text{"kgf/cmq.g"} \\ \text{"psi.a"} \\ \text{"psi.g"} \\ \text{"atm.a"} \\ \text{"atm.g"} \\ \text{"mmH2O.a"} \\ \text{"mmH2O.g"} \\ \text{"inH2O.a"} \\ \text{"inH2O.g"} \\ \text{"mmHG.a"} \\ \text{"mmHG.g"} \end{bmatrix}$$

mc_getP(*stream*) Pa = 290.0755 psi multiply by current Prode pressure unit, then request any unit in the result

sel := 13 select a new pressure unit

mc_setUMC(*UM*, *sel*) = 1 change to the 11th unit for pressure mc_getSUMS(*UM*) = "psi.a" show current unit name for UM

mc_getP(*stream*) psi = 290.0804 psi now pressure results must be multiplied by psi

mc_setUMC(*UM*, 1) = 1 reset to original unit for remainder of worksheet

Routines UMCR, UMCS, and UMAU are not fully documented in the Prode manual so they have been left out of the dll.

mc_UMRAU(*UM*) = 1 removes all added units for (property no.)

Error message flags

last error message, maybe from a previous run

mc_ErrMsg(" ") = "Error accessing component's data archive"

dum := "dummy"

errflag := mc_getErrFlag(*dum*) 0 = no errors, 1 = errors found

errflag = 1

This flag only works if the Window Dialog messages are turned off.Otherwise, the Dialog messages are themselves the indication of errors.See mc_defErrMsg.Errors that Mathcad detects (i.e. red indication) are not included for this flag.

At the time this test file was created, the thermal conductivity of solid water was not available in the database, causing an error and a value of 1 for errflag.

mc_setErrFlag(0) = 0 set to 0at start of next calc's to clear flag

mc_defErrMsg(0) = ■

0 = turns off the Window Dialog messages
1 = turns on the Window Dialog messages

This function was demonstrated at start of worksheet.Turning off the Window Dialog messages allows the computations to continue without pausing to close the Dialog window when an error occurs.The error may still be visible if a 0 value is returned where a real number is expected.

Atmospheric pressure

patm := mc_getPatm("mc") = $1.0133 \cdot 10^{5}$ the pressure shouldbe 1.013105

Base values for enthalpy and entropy

The default values for the base temperature, enthalpy, and entropy may be found in the config>settings Prode window. The functions below may be used to change the settings. The settings apply only to the current archive. The settings for the archive are saved when the archive is saved.

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

If code = 2, the tref and val inputs are ignored.

$code := 1$ $tref := 298$ $val := 0$

`mc_setHB(code, tref, val) = 1`   enthalpy references

`mc_setSB(code, tref, val) = 1`   entropy references

Read/write stream properties

If a write operation exists, it will appear under the read operation, using the value from the read operation.This simplifies the testing process.

The write operations in this section are in blue highlight.

$stream := 1$

$phase := 2$ the phase position (not the phase type)

$cpos := 2$ cpos is the component's numerical position in the composition vector for the stream, starting with 1

`mc_isSDef(stream) = 1`   given a stream returns TRUE (integer = 1) if stream has been defined, otherwise returns FALSE (0)

`name := mc_StrN(stream) = "Test Case 1"`   stream name

`mc_putN(stream, name) = 1`

`mc_setOp(stream, 150, patm) = 1`   This is an edit operation to lower the temperature so liquid will be present for the functions below.

`t := mc_getT(stream) = 150`   temperature

`mc_putT(stream, t) = 1`

`p := mc_getP(stream) =` $1.0133 \cdot 10^{5}$ pressure

`mc_putP(stream, p) = 1`

`pnr := mc_getPNr("mc") = 10`   returns the maximum number of phases that procedure can detect in the archive for all streams (may include phases at other temperatures)

`mc_StrPt(stream, phase) = 2`   given a stream and phase # in range 1- getPNr() returns the phase type (0=vapor,1=liquid,2=solid)

`i_ := [1..pnr]` $stream = 1$

$phases_{i\_}$ `:= mc_StrPts(stream, i_)`   given a stream and phase # in range 1- getPNr() returns a ANSI C string with the description of type for detected phase

$$phases = \begin{bmatrix} \text{"Vapor"} \\ \text{"Solid"} \\ \text{"Solid"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \end{bmatrix}$$

only one liquid phase is present
later, the flash routine code will be reset to obtain two liquid phases

mc_StrLf(*stream*) = 0	given a stream returns the total liquid fraction (molar basis) in stream
mc_StrPf(*stream*, *phase*) = 0.1456	given a stream and phase phase # in range 1- getPNr() returns the phase fraction
w := mc_getW(*stream*, *phase*, *cpos*) = $1.0251 \cdot 10^{-6}$	mole fraction of component (cpos #) in a phase
mc_putW(*stream*, *phase*, *cpos*, *w*) = 1	mole fraction w of component cpos in a phase
rate := mc_getWm(*stream*) = 1	stream flow rate, mass/time
mc_setWm(*stream*, *rate*) = 1	
zi := mc_getZ(*stream*, *cpos*) = 0.15	mole fraction of component cpos in total stream
mc_putZ(*stream*, *cpos*, *zi*) = 1	
mc_getCNr(*stream*) = 3	number of components in stream
mc_StrZv(*stream*) = 0.9841	returns the relevant compressibility factor (gas phase)
mc_StrMw(*stream*) = 22.9437	molecular weight of total stream
mc_StrGMw(*stream*) = 16.4078	molecular weight of gas phase
mc_StrLMw(*stream*) = 0	molecular weight of liquid phase
mc_StrV(*stream*) = 0	specific volume as sum of specific volumes of all phases

enthalpy

h := mc_StrH(*stream*) = −526.8941	total stream enthalpy
mc_StrGH(*stream*) = −157.4136	gas phase enthalpy
mc_StrSGH(*stream*) = −309.6676	gas specific enthalpy
mc_StrLH(*stream*) = 0	liquid enthalpy
mc_StrSLH(*stream*) = 0	liquid specific enthalpy
mc_StrSH(*stream*) = −369.4805	solid enthalpy

mc_StrSSH(*stream*) = −751.482	solid specific enthalpy

entropy

entropy := mc_StrS(*stream*) = −2.5474	total stream entropy
mc_StrGS(*stream*) = −0.7266	gas phase entropy
mc_StrSGS(*stream*) = −1.4294	gas specific entropy
mc_StrLS(*stream*) = 0	liquid entropy
mc_StrSLS(*stream*) = 0	liquid specific entropy
mc_StrSS(*stream*) = −1.8208	solid entropy
mc_StrSSS(*stream*) = −3.7034	solid specific entropy

heat capacity, mass basis

mc_StrGICp(*stream*) = 2.0277	ideal gas heat capacity
mc_StrGCp(*stream*) = 2.0551	gas constant pressure heat capacity
mc_StrGCv(*stream*) = 1.523	gas constant volume heat capacity
mc_StrLCp(*stream*) = 0	liquid constant pressure heat capacity
mc_StrLCv(*stream*) = 0	liquid constant volume heat capacity
mc_StrSCp(*stream*) = 1.36	solid constant pressure heat capacity

speed of sound

mc_StrMSS(*stream*) = ∎	mixed phase speed of sound HEM model
mc_StrGSS(*stream*) = 266.4921	gas phase
mc_StrLSS(*stream*) = 0	liquid phase

Joule Thomson coefficient

mc_StrGJT(*stream*) = $1.4851 \cdot 10^{-5}$	gas phase
mc_StrLJT(*stream*) = 0	liquid phase

compressibility, expansivity

mc_StrGIC(*stream*) = $1.003 \cdot 10^{-5}$	gas isothermal compressibility, 1VPV
mc_StrLIC(*stream*) = 0	liquid isothermal compressibility
mc_StrGVE(*stream*) = 0.0069	gas volumetric expansivity 1VTV
mc_StrLVE(*stream*) = 0	gas volumetric expansivity

density

$\mathrm{mc\_StrGD}(stream)=1.3546$	gas density
$\mathrm{mc\_StrLD}(stream)=0$	liquid density

thermal conductivity

$\mathrm{mc\_StrGC}(stream)=0.0159$	gas conductivity
$\mathrm{mc\_StrLC}(stream)=0$	liquid conductivity

viscosity

$\mathrm{mc\_StrGV}(stream)=6.0897\cdot 10^{-6}$	gas viscosity
$\mathrm{mc\_StrLV}(stream)=0$	liquid viscosity

surface tension

$\mathrm{mc\_StrST}(stream)=0$	liquid/gas

flammability

$\mathrm{mc\_StrFML}(stream)=4.995$	gas phase lean limit
$\mathrm{mc\_StrFMH}(stream)=15.0622$	gas phase rich limit

other stream properties

$\mathrm{mc\_StrHC}(stream)=48365.4562$	gas phase heat of combustion
$compcode:=\mathrm{mc\_getCC}(stream, cpos)=594$	component number for component = cpos
$\mathrm{mc\_putCC}(stream, cpos, compcode)=1$	
$\mathrm{mc\_getMCNr}(\text{"dummy"})=50$	maximum number of components in a stream

interactions

$int\_pos:=1$	the interaction number (i.e. the row in the BIP window for the stream)
$\mathrm{mc\_getMBPNr}(\text{"dummy"})=250$	maximum number of binary pairs in a stream (for all streams)
$ci:=\mathrm{mc\_getCi}(stream, int\_pos)=1$	component index i in interaction list
$\mathrm{mc\_putCi}(stream, int\_pos, ci)=1$	
$cj:=\mathrm{mc\_getCj}(stream, int\_pos)=2$	component index j in interaction list
$\mathrm{mc\_putCj}(stream, int\_pos, cj)=1$	
$model:=\mathrm{mc\_getMB}(stream, int\_pos)=50$	returns model number for interaction # int_pos in given stream

`mc_putMB(stream, int_pos, model) = 1` sets the model number for a given stream and interaction number, int_pos

$id := 0$ This id is the BIP column (starting with 0 for BIP-1) shown in the PPP window under BIPs for the stream.

`Kji := mc_getBIP(stream, int_pos, id) = 0.1183` value of the interaction coefficient

`mc_putBIP(stream, int_pos, id, Kji) = 1` specifies a value for an interaction coefficient

thermodynamic models for streams

$stream = 1$

$Kcode := 50$ code for SRK standard model package, see manual for other package numbers

`mc_setKM(stream, Kcode) = 1`

mp codes for functions below

1fugacity
2enthalpy
3entropy
4molar volume
5viscosity

Examples for fugacity and enthalpy models below:

$state := 0$ vapor state

$mp := 1$ fugacity model for stream, state

`fmodel := mc_getMP(stream, mp, state) = 50` retrieve model number

`mc_setMP(stream, mp, fmodel, state) = 1` set model number

$mp := 2$ enthalpy model for stream, state

`hmodel := mc_getMP(stream, mp, state) = 50` retrieve model number

`mc_setMP(stream, mp, hmodel, state) = 1` set model number

Thermodynamic calculations

$stream := 5$ use stream 5 for examples below

$t = 150$ $p = 1.0133 \cdot 10^{5}$

$state := 1$ state (0=vapor, 1=liquid, 2=solid)

phase equilibria

$n := 1$ $pf := .3$	see below
mc_PfPF(*stream*, *p*, *pf*, *state*, *n*) = 290.766	n th equilibrium temp at p, pf (phase fraction), state (0=vapor, 1=liquid, 2=solid)
mc_PfTF(*stream*, t, *pf*, *state*, *n*) = 0	n th equil. press at t, pf, state
$lf := .2$	set liquid fraction
mc_LfPF(*stream*, *p*, *lf*) = 299.6111	first equil. temp at liquid fraction, lf
mc_LfTF(*stream*, t, *lf*) = 0	first equil. pressure at liquid fraction, lf
mc_StrCPnr(*stream*) = 1	number of critical points found
$cpn := 1$	selected critical point
mc_StrPc(*stream*, *cpn*) = $4.6926 \cdot 10^{6}$	critical pressure for critical point #, cpn
mc_StrCBp(*stream*) = $4.6934 \cdot 10^{6}$	cricondenBar pressure
mc_StrCBt(*stream*) = 444.1406	cricondenBar temperature
mc_StrCTp(*stream*) = $4.4361 \cdot 10^{6}$	cricondenTherm pressure
mc_StrCTt(*stream*) = 446.9182	cricondenTherm temperature
mc_StrAc(*stream*) = 0.2077	acentric factor (mole fraction average)

phase diagrams

$stream := 5$

lnr := mc_PELNr(*stream*) — Given a stream calculates the phase diagram and returns the number of equilibrium lines available

$lnr = 2$

line types

$line := 2$

ltype := mc_PELT(*stream*, *line*) — Given a stream and line number, returns the line type:
bubble line
dew line
three phase line
fractional phase

$ltype = 2$

line properties

$lprop := \mathrm{mc\_PELP}(stream, line)$

Given a stream and line, returns the line properties:
vapor-liquid
vapor-liquid-liquid
vapor-solid
liquid-solid
fractional phase

$lprop = 1$

equilibrium lines
The prode.dll has assumed a maximum number of points of 50 for the equilibrium lines.This dimension cannot be changed dynamically for the variables passed to and from Mathcad.Therefore, the mc_PELine routine leaves out the maxpt variable that is shown in the Prode corresponding routine.

The mc_PELine function (see the first line in the program below) produces a matrix result.Although this matrix can be used "as is" the Mathcad program, "PELine" below calls mc_PELine and splits the matrix into the separate variables.

$stream = 5$

```
PELine(streamx, linex) := | M := mc_PELine(streamx, linex)
                          | npts := M
                          |          1 3
                          | Te := submatrix(M, 1, npts, 1, 1) K
                          | Pr := submatrix(M, 1, npts, 2, 2) Pa
                          | [ Te  Pr  npts ]
```

Given stream and equilibrium line number, the temperature and pressure vectors and the total number of points are computed and returned.

The output is shown below.

$[\,T1 \;\; P1 \;\; npts1\,] := PELine(stream, line)$

$npts1 = 31$

$$T1 = \begin{bmatrix} 311.1738\ \mathrm{K} \\ 316.4703\ \mathrm{K} \\ 322.564\ \mathrm{K} \\ 327.6265\ \mathrm{K} \\ 332.689\ \mathrm{K} \\ 337.7515\ \mathrm{K} \\ 342.814\ \mathrm{K} \\ 347.8765\ \mathrm{K} \\ 356.6734\ \mathrm{K} \\ 361.6734\ \mathrm{K} \\ 366.6734\ \mathrm{K} \\ 371.6734\ \mathrm{K} \\ 376.6734\ \mathrm{K} \\ 381.6734\ \mathrm{K} \\ 386.6734\ \mathrm{K} \\ \vdots \end{bmatrix} \qquad P1 = \begin{bmatrix} 1.0133 \cdot 10^{5}\ \frac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.2358 \cdot 10^{5}\ \frac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.5389 \cdot 10^{5}\ \frac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.8338 \cdot 10^{5}\ \frac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 2.1723 \cdot 10^{5}\ \frac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 2.559 \cdot 10^{5}\ \frac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ \vdots \end{bmatrix}$$

phase fraction lines

$stream := 5$
$state := 0$

$fraction := .5$

```
PFLine(stream, state, fraction) := | M := mc_PFLine(stream, state, fraction)
                                   | npts := M_{1 3}
                                   | Tem := submatrix(M, 1, npts, 1, 1) K
                                   | Pr := submatrix(M, 1, npts, 2, 2) Pa
                                   | [ Tem Pr npts ]
```

Given stream, state, and fraction of that state, computes the temperature and pressure vectors along that phase ction, plus the number of points on the curve.

$[\,Tf\ \ Pf\ \ nf\,] := PFLine(stream, state, fraction)$

$nf = 35$

$$Tf = \begin{bmatrix} 270.8829\ \mathrm{K} \\ 275.8829\ \mathrm{K} \\ 280.8829\ \mathrm{K} \\ 285.8829\ \mathrm{K} \\ 290.8829\ \mathrm{K} \\ 295.8829\ \mathrm{K} \\ 300.8829\ \mathrm{K} \\ 305.8829\ \mathrm{K} \\ 310.8829\ \mathrm{K} \\ 315.8829\ \mathrm{K} \\ 320.8829\ \mathrm{K} \\ 325.8829\ \mathrm{K} \\ 330.8829\ \mathrm{K} \\ \vdots \end{bmatrix} \qquad Pf = \begin{bmatrix} 1.0133\cdot 10^{5}\ \dfrac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.2093\cdot 10^{5}\ \dfrac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.4338\cdot 10^{5}\ \dfrac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.6895\cdot 10^{5}\ \dfrac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ 1.9791\cdot 10^{5}\ \dfrac{\mathrm{kg}}{\mathrm{m\,s}^{2}} \\ \vdots \end{bmatrix}$$

$linef := \mathrm{augment}(Tf, Pf)$

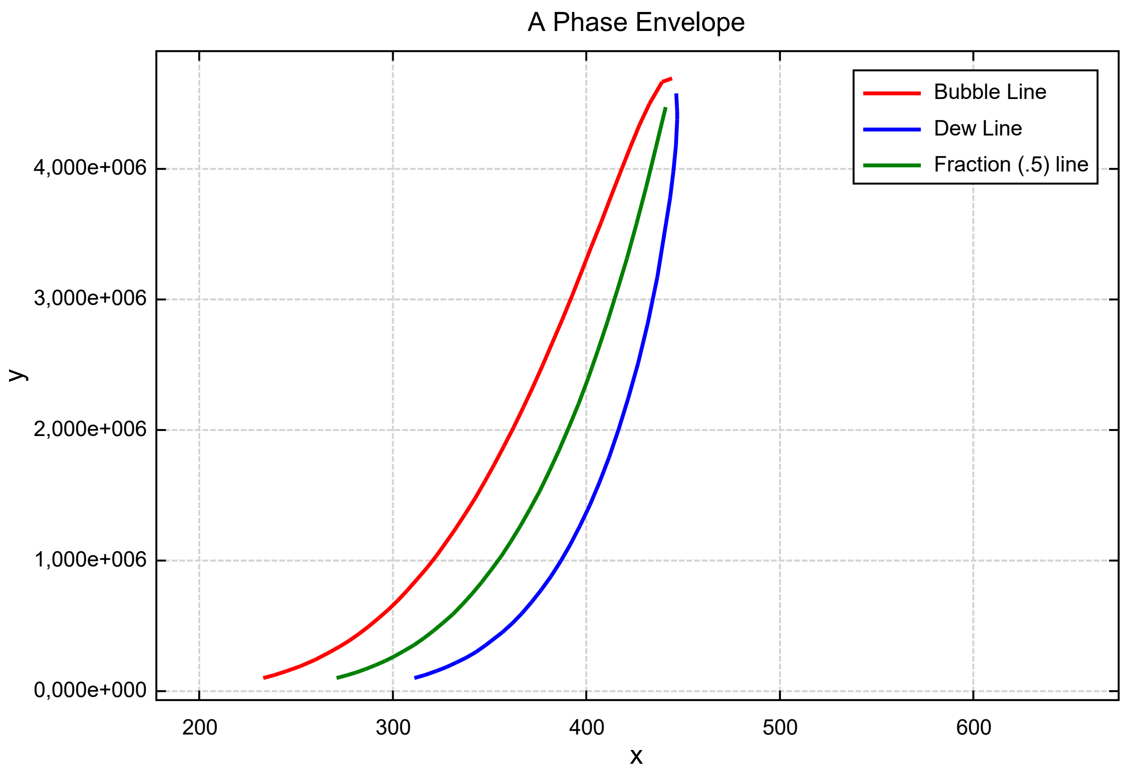
$lnr := \mathrm{mc\_PELNr}(stream) = 2$

```
for line ∈ [1..lnr]
    [ Tl_line  Pl_line  nl_line ] := PELine(stream, line)
    typel_line := mc_PELT(stream, line)
    propl_line := mc_PELP(stream, line)
    linexy_line := augment(Tl_line, Pl_line)
```

```
The lnr statement and the code
on the left replaces
the PhaseEnv program in the
Mathcad version of this file.
```

$$Tl_1 = \begin{bmatrix} 232.9657\ K \\ 238.599\ K \\ 244.6927\ K \\ 252.2865\ K \\ 259.8802\ K \\ 267.474\ K \\ 272.474\ K \\ 277.474\ K \\ 282.474\ K \\ \vdots \end{bmatrix} \quad Pl_1 = \begin{bmatrix} 1.0133 \cdot 10^5\ \frac{kg}{m\ s^2} \\ 1.2342 \cdot 10^5\ \frac{kg}{m\ s^2} \\ 1.5124 \cdot 10^5\ \frac{kg}{m\ s^2} \\ 1.9221 \cdot 10^5\ \frac{kg}{m\ s^2} \\ \vdots \end{bmatrix} \quad Tl_2 = \begin{bmatrix} 311.1738\ K \\ 316.4703\ K \\ 322.564\ K \\ 327.6265\ K \\ 332.689\ K \\ 337.7515\ K \\ 342.814\ K \\ 347.8765\ K \\ 356.6734\ K \\ 361.6734\ K \\ \vdots \end{bmatrix} \quad Pl_2 = \begin{bmatrix} 1.0133 \cdot 10^5\ \frac{kg}{m\ s^2} \\ 1.2358 \cdot 10^5\ \frac{kg}{m\ s^2} \\ 1.5389 \cdot 10^5\ \frac{kg}{m\ s^2} \\ 1.8338 \cdot 10^5\ \frac{kg}{m\ s^2} \\ \vdots \end{bmatrix}$$

As shown in the nc vector, the lines may have different number of points.In order to prevent curves returning to the origin, extract the data from Tj and Pj.



$$\begin{cases} linexy_1 \\ linexy_2 \\ linef \end{cases}$$

The PELine and PFLine programs and the code for the phase envelope curves may be copied into or referenced by other programs.

hydrates

$hydmodel := 1$ $\quad thyd := 260$

$str\_hyd := 2$ stream for hydrate function below

hydmodel =
1 = assume free water present, this option produces conservative but safe values
2 = calculate amount of water in liquid phase
3 = solve as multiphase equilibria, solve phase equilibria including solids as ice

Since water is not present in the stream chosen for testing, the 2 and 3 hydmodels will return 0.

$phyd := \mathrm{mc\_HPFORM}(str\_hyd, thyd, hydmodel) = 0$ returns the pressure that hydrates form at temperature = thyd

The HTFORM Prode function is not available in the Basic version, but the HPFORM function should suffice.

flashes

$stream = 5$

$\mathrm{mc\_setSOp}(stream) = 1$ flash at standard conditions

$et := 0$ estimated temperature set to 0 for automatic

$ep := 0$ estimated pressure set to 0 for automatic

$\mathrm{mc\_setOp}(stream, 150, patm) = 1$ set new operating conditions and flash

$t := \mathrm{mc\_getT}(stream) = 150$ temperature

$p := \mathrm{mc\_getP}(stream) = 1.0133 \cdot 10^{5}$ pressure

$h := \mathrm{mc\_StrH}(stream) = -811.9593$ enthalpy obtained above

$entropy := \mathrm{mc\_StrS}(stream) = -3.281$ entropy obtained above

$sv := \mathrm{mc\_StrV}(stream) = 0.0014$ volume obtained above

find temperature

$\mathrm{mc\_VPF}(stream, p, sv, et) = 150$ volume-pressure flash, et=temp guess

$\mathrm{mc\_HPF}(stream, p, h, et) = 150$ enthalpy-pressure flash, et=temp guess

$\mathrm{mc\_SPF}(stream, p, entropy, et) = 150$ entropy-pressure flash, et=temp guess

find pressure

$\mathrm{mc\_VTF}(stream, t, sv, ep) = 0$ volume-temp flash, ep=press guess

$\mathrm{mc\_HTF}(stream, t, h, ep) = 0$ enthalpy-temp flash, ep=press guess

$\mathrm{mc\_STF}(stream, t, entropy, ep) = 0$ entropy-temp flash, ep=press guess

The flashes that determine pressure have some difficulty converging for multiphase (liquids and solids) problems.Select another flash routine and iterate if needed.

Extended methods for accessing stream properties

These functions allow simultaneous setting of temperature and pressure followed by an isothermal flash before the desired property is returned.These methods should be used with care because of the change in the stream conditions.

stream := 2	
mc_EStrGMw(stream, t, p) = 16.0474	gas molecular weight
mc_EStrLMw(stream, t, p) = 55.7928	liquid molecular weight
mc_EStrLf(stream, t, p) = 0.0009	mole fraction of liquid
phase := 1	position of phase, not the state code. positions are usually vapor=1, liquid=2, solid=3 but extra liquid and solid phases may be present
mc_EStrPf(stream, phase, t, p) = 0.9991	molar phase fraction of phase
mc_EStrZv(stream, t, p) = 0.9843	gas (vapor) compressibility factor
mc_EStrH(stream, t, p) = −317.8313	total enthalpy
mc_EStrV(stream, t, p) = 0.7525	total specific volume
mc_EStrGCp(stream, t, p) = 2.103	gas constant pressure heat capacity
mc_EStrGCv(stream, t, p) = 1.5593	gas constant volume heat capacity
mc_EStrLCp(stream, t, p) = 1.7035	liquid constant pressure heat capacity
mc_EStrLCv(stream, t, p) = 1.3968	liquid constant volume heat capacity
mc_EStrGIC(stream, t, p) = $1.0028 \cdot 10^{-5}$	gas isothermal compressibility
mc_EStrLIC(stream, t, p) = $5.1467 \cdot 10^{-10}$	liquid isothermal compressibility
mc_EStrMSS(stream, t, p) = 317.7297	mixture speed of sound
mc_EStrGSS(stream, t, p) = 318.2669	gas speed of sound
mc_EStrLSS(stream, t, p) = 3354.1111	liquid speed of sound
mc_EStrGJT(stream, t, p) = $1.4662 \cdot 10^{-5}$	gas Joule Thomson coefficient
mc_EStrLJT(stream, t, p) = $-6.9784 \cdot 10^{-7}$	liquid Joule Thomson coefficient
mc_EStrGVE(stream, t, p) = 0.0069	gas volumetric expansivity coefficient
mc_EStrLVE(stream, t, p) = 0.0009	liquidvolumetric expansivity coefficient
mc_EStrHC(stream, t, p) = 50008.9081	heat of combustion
mc_EStrFML(stream, t, p) = 4.999	lean flammability limit of gas
mc_EStrFMH(stream, t, p) = 14.9987	rich flammability limit of gas
mc_EStrS(stream, t, p) = −1.4661	total entropy

$\mathrm{mc\_EStrGD}(stream, t, p) = 1.3246$	gas density
$\mathrm{mc\_EStrLD}(stream, t, p) = 730.588$	liquid density
$\mathrm{mc\_EStrGC}(stream, t, p) = 0.0161$	gas thermal conductivity
$\mathrm{mc\_EStrLC}(stream, t, p) = 0.1754$	liquid thermal conductivity
$\mathrm{mc\_EStrGV}(stream, t, p) = 6.0282 \cdot 10^{-6}$	gas viscosity
$\mathrm{mc\_EStrLV}(stream, t, p) = 0.0013$	liquid viscosity
$\mathrm{mc\_EStrST}(stream, t, p) = 0.0288$	surface tension

Fugacity and derivatives

The operations below behave like subroutines rather than functions because they return more than one result. The Mathcad system imposes some restrictions on function input and output so the normal C++ methods of passing variables is not possible.These restrictions are needed to enforce the "non code" look of the Mathcad interface .As a result of these restrictions, the functions below have slightly different argument lists than found in Prode and all of the results are returned in a single matrix .Mathcad routines are then provided to split these results into the appropriate variables.

The prode.dll has assumed a maximum number of components of 50 for all vector and matrix routines.This dimension cannot be changed dynamically for the variables passed to and from Mathcad .For greater number of components, prode.dll must be rebuilt. In that case, the constant "maxnc" in the source code for the routines in this section must be changed to the higher number.

$stream := 1$

$NC := \mathrm{mc\_getCNr}(stream)$

$t = 150$ These variables were defined above.

$p = 1.0133 \cdot 10^{5}$

$\mathrm{mc\_setOp}(stream, t, p) = 1$

$i\_ := [1 .. NC]$

$phase := 2$

$mf_{i\_} := \mathrm{mc\_getW}(stream, phase, i\_)$ mf = mole fractions of components is stream

$$mf_{i\_} = \begin{bmatrix} 1.5711 \cdot 10^{-7} \\ 1.0251 \cdot 10^{-6} \\ 1 \end{bmatrix}$$

$state := 1$ The liquid state is being used.

$process := 1$ Up to 5 processes may be defined in the base Prode version. Processes may be redefined with new streams.

$\mathrm{mc\_DPinit}(process, stream) = 1$

fugacity vector

$fg := \mathrm{mc\_StrFv}(process, state, t, p, mf, NC)\ \mathrm{Pa}$ This routine returns the fugacity vector alone.

$$fg = \begin{bmatrix} 2.2571 \cdot 10^{7} \dfrac{\mathrm{kg}}{\mathrm{m\ s}^{2}} \\ 55244.7405 \dfrac{\mathrm{kg}}{\mathrm{m\ s}^{2}} \\ 1077.6956 \dfrac{\mathrm{kg}}{\mathrm{m\ s}^{2}} \end{bmatrix}$$

$fugacity_{i\_} := fg_{i\_} \cdot mf_{i\_}$

The Prode routines define the "fugacity" variable as the fugacity coefficient times the total pressure. Thus, fugacity is obtained by the equation on the left.

$$fugacity = \begin{bmatrix} 3.5463 \dfrac{\mathrm{kg}}{\mathrm{m\ s}^{2}} \\ 0.0566 \dfrac{\mathrm{kg}}{\mathrm{m\ s}^{2}} \\ 1077.6943 \dfrac{\mathrm{kg}}{\mathrm{m\ s}^{2}} \end{bmatrix}$$

fugacity vector plus derivatives wrt T, P, w

```
StrFvd(process, state, t, p, mf, NC) := | M := mc_StrFvd(process, state, t, p, mf, NC)
                                        | "separate the results into vectors and a matrix"
                                        | fg := submatrix(M, 1, NC, 1, 1)
                                        | dfgt := submatrix(M, 1, NC, 2, 2)
                                        | dfgp := submatrix(M, 1, NC, 3, 3)
                                        | dfgmf := submatrix(M, 1, NC, 4, 4 + NC - 1)
                                        | [ fg dfgt dfgp dfgmf ]
```

1given the stream, state, temp, press, composition vector, w,and the number of components, NC, return fugacity vector, fg, and derivatives of fg wrt t, p, and mf.

$[\ fg\ dfgt\ dfgp\ dfgmf\ ] := StrFvd(process, state, t, p, mf, NC)$

the default Prode units are applied to the results below but the variables are kept dimensionless for use in Prode Physical Properties program.

$$fg\ \mathrm{Pa} = \begin{bmatrix} 2.2571 \cdot 10^{7} \\ 55244.7405 \\ 1077.6956 \end{bmatrix} \mathrm{Pa} \qquad dfgt\ \frac{\mathrm{Pa}}{\mathrm{K}} = \begin{bmatrix} 3.5879 \cdot 10^{5} \\ 4372.2585 \\ 115.1895 \end{bmatrix} \frac{\mathrm{Pa}}{\mathrm{K}} \qquad dfgp = \begin{bmatrix} 0.6059 \\ 0.0014 \\ 2.6093 \cdot 10^{-5} \end{bmatrix}$$

$$dfgmf\ \mathrm{Pa} = \begin{bmatrix} -3.7021 \cdot 10^{8} & -2.841 \cdot 10^{8} & -2.2457 \cdot 10^{8} \\ -3.6321 \cdot 10^{5} & -4.6135 \cdot 10^{5} & -2.1749 \cdot 10^{5} \\ 0.01 & 0.0114 & 0.006 \end{bmatrix} \mathrm{Pa}$$

Other stream state variables and their derivatives

Functions were provided above (eg. mc_StrH)to obtain the enthalpy (H), entropy(S), and molar volume (V) of a stream. The next routine allows the operating conditions (t, p, w) to be specified to values other than those in the stream data file. The user selects which variable, H, S, or V, is desired, using a string variable with the corresponding variable initial. The program calls the appropriate mc_xxx function and then separates the variables from the output matrix.

```
StrXvd(X, process, state, t, p, mf, NC) := | if X = "H"
                                           |    M := mc_StrHvd(process, state, t, p, mf, NC)
                                           | else
                                           |    "do nothing"
                                           | if X = "S"
                                           |    M := mc_StrSvd(process, state, t, p, mf, NC)
                                           | else
                                           |    "do nothing"
                                           | if X = "V"
                                           |    M := mc_StrVvd(process, state, t, p, mf, NC)
                                           | else
                                           |    "do nothing"
                                           | x := col(M, 1)
                                           | dxt := col(M, 2)
                                           | dxp := col(M, 3)
                                           | dxmf := submatrix(M, 1, 1, 4, 4 + NC - 1)
                                           | [ x dxt dxp dxmf ]
```

$[H\ dHt\ dHp\ dHmf] := StrXvd(\text{"H"}, process, state, t, p, mf, NC)$

display results with default Prode units

$$H \frac{\text{kJ}}{\text{kmol}} = [-24964.9683] \frac{\text{kJ}}{\text{kmol}}$$

$$dHt \frac{\frac{\text{kJ}}{\text{kmol}}}{\text{K}} = [59.2059] \frac{\text{kJ}}{\text{kmol K}}$$

$$dHp \frac{\frac{\text{kJ}}{\text{kmol}}}{\text{Pa}} = [2.4984 \cdot 10^{-5}] \frac{\text{kJ}}{\text{kmol Pa}}$$

$$dHmf \frac{\text{kJ}}{\text{kmol}} = [-8000.1235\ -19802.456\ -24959.9131] \frac{\text{kJ}}{\text{kmol}}$$

$[S\ dSt\ dSp\ dSw] := StrXvd(\text{"S"}, process, state, t, p, mf, NC)$

$$S \frac{\text{kJ}}{\text{kmol K}} = [-118.5426] \frac{\text{kJ}}{\text{kmol K}}$$

$$dSt \frac{\text{kJ}}{\text{kmol K}^2} = [0.3947] \frac{\text{kJ}}{\text{kmol K}^2}$$

$$dSp\ \frac{kJ}{kmol\ K\ Pa}=\left[-3.4749\cdot 10^{-8}\right]\frac{kJ}{kmol\ K\ Pa}$$

$$dSw\ \frac{kJ}{kmol\ K}=\left[-71.4065\ \ -99.9233\ \ -101.9207\right]\frac{kJ}{kmol\ K}$$

$$\left[V\ dVt\ dVp\ dVmf\right]:=StrXvd\left(\text{"V"},process,state,t,p,mf,NC\right)$$

$$V\ \frac{m^3}{mol}=\left[0.0302\ \frac{m^3}{mol}\right]$$

$$dVt\ \frac{m^3}{mol\ K}=\left[3.4749\cdot 10^{-5}\ \frac{m^3}{mol\ K}\right]$$

$$dVp\ \frac{m^3}{mol\ Pa}=\left[-1.0646\cdot 10^{-11}\right]\frac{m^3}{mol\ Pa}$$

$$dVmf\ \frac{m^3}{mol}=\left[0.0637\ \ 0.0614\ \ 0.0604\right]\frac{m^3}{mol}$$

Operations to set/retrieve the options needed for equation of state models and flash routine phases

See the Prode manual (see paragraph "Codes used in Prode library") and also open the Prode drop menus for the model to view the description of the options set by the OM code variable. The user will probably find it easier to set the options using the Prode window.

All even code values mean that only single liquid phases are allowed in the flash routines. For multiple liquids, the code value must be odd.

$stream:=1$

$option:=\mathrm{mc\_getOM}(stream)=554$

current option set
This should = 552 for stream 1 in the def.ppp archive.
With this option value, only L-V flashes will result.

$\mathrm{mc\_setOM}(stream, option+1)=1$

This changes the flashes of "stream" to multiple liquids.

$\mathrm{mc\_setOp}(stream, t, p)=1$

redo the flash

given a stream and phase # in range 1- getPNr() returns the phase type (0=vapor,1=liquid,2=solid)

$i\_:=[1..pnr]$

given a stream and phase # in range 1- getPNr() returns a ANSI C string with the description of type for detected phase

$phase_{i\_}:=\mathrm{mc\_StrPts}(stream, i\_)$

$$phases = \begin{bmatrix} \text{"Vapor"} \\ \text{"Solid"} \\ \text{"Solid"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \\ \text{"Not present"} \end{bmatrix}$$

Previously, only one liquid was present

mc_setOM(*stream*, *option*) = 1 — reset the code to single liquids

Initializing a stream

This section was changed for this version.

The example will create a stream with water and methanol. The component numbers in the Prode databank can change with updates, so always use CAS numbers when initializing by program instead of manually using the Prode window.

```
methanol_id := 67561                                   CAS number of methanol
methanol_code := mc_CompCID(methanol_id) = 1047
water_id := 7732185                                    CAS number of water
water_code := mc_CompCID(water_id) = 1631
stream := 11
mc_initS(stream) = 1                                   initialize a new stream
model := 50      SRK standard                          see Prode manual for model codes
mc_setKM(stream, model) = 1                            set property model package
mc_putZ(stream, 1, .5) = 1                             set total stream mole fractions
mc_putZ(stream, 2, .5) = 1
mc_putCC(stream, 1, methanol_code) = 1                 define components
mc_putCC(stream, 2, water_code) = 1
mc_setS(stream) = 1                                    validate the stream

                                                        codeeq =0 for VLE
 codeeq := 0                                                   1 for LLE
                                                               2 for SLE
                                                               3 for hydrates

mc_loadSB(stream, codeeq) = 1                          load BIP coefficients
mc_setWm(stream, 1.3) = 1                              set mass flow rate
temp := 300      pres := patm
mc_setOp(stream, temp, pres) = 1                       set temp and pres and flash
mc_edS(stream) = 0                                     view the stream then press OK
```

Other stream operations

$stream2 := 1$

$stream1 := 10$

mc_StrCopy(*stream1* , *stream2*) = 1 — copy stream2 to stream1 (note the order!)

$et :=$ mc_getT(*stream2*) = 150

mc_getT(*stream1*) = 150

press := mc_getP(*stream1*) $= 1.0133 \cdot 10^{5}$ — *stream_out* := 9 — *et* = 150

mc_MixF(*stream_out* , *stream1* , *stream2* , *press* ·.9 , *et*) = 1 — flash at lower stream press, et=temp guess for mixed stream..

mc_getT(*stream_out*) = 149.5844 — mixed stream temperature

$stream2 := 12$ — the new stream to be created by Divi

$wdiv := .7$

mc_Divi(*stream1* , *stream2* , *wdiv*) = 1 — Given one stream (stream1) 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)

Only one new stream is created, NOT two.The starting stream gets overwritten as shown above.

phase separation

$stream1 := 5$

$stream2 := 13$ — the new stream to be created by PSep

$phase := 1$ — phase number to separate, NOT the phase type

mc_PSep(*stream1* , *stream2* , *phase*) = 1 — Given a stream (stream1) performs an isothermal flash to simulate a phase separator and returns the specified phase number (not phase type) to stream2.

$gasstream := 14$

mc_GSep(*stream1* , *gasstream*) = 1 — Given a stream (stream1) performs an isothermal flash to simulate a phase separator and returns the gas phase to gasstream

$liqstream := 15$

mc_LSep(*stream1* , *liqstream*) = 1 — Given a stream (stream1) performs an isothermal flash to simulate a phase separator and returns the liq phase(s) to liqstream

Polytropic compressor/expander

rate compressor efficiency

$pin := 10^{6}$ pressure in Pa

$pout := 2 \cdot 10^{6}$

$tin := 300$ temperature in K

$tout := 370$

$model := 2$

$stream := 2$

for a rating, model may be the following:
2 = Huntington method
4 = Paron method

mc_setOp$(stream, tin, pin) = 1$ set the inlet stream conditions

mc_PSPF$(stream, pout, model, tout) = 0.745$ efficiency rating and "stream" in archive now contains the outlet conditions

design model

$eff := .75$ polytropic efficiency given

$model := 1$

for a design, model may be the following:
1 = Huntington
3 = Paron

mc_setOp$(stream, tin, pin) = 1$ reset the inlet stream conditions

mc_PSPF$(stream, pout, model, eff) = 369.2757$ outlet temperature and "stream" now contains the outlet conditions

Isentropic expansion, nozzles

$stream := 5$

$tin := 340$ $pin := 2 \cdot 10^{6}$

mc_setWm$(stream, 1.23) = 1$

mc_setOp$(stream, tin, pin) = 1$ set stream conditions

$model := 2$

model options::
1 = homogeneous, equilibrium
2 = homogeneous, non equilibrium
3 = homogeneous, non equilibrium ?
4 = nonhomogeneous, non equilibrium

$pout := patm$

$parameter := .75$ Prode manual does not explain this parameter

mc_ISPF$(stream, pout, model, parameter) = 4.2071 \cdot 10^{-5}$ calculated orifice area, m2

mc_getErrFlag$(" ") = 1$

Pipe flow

The PIPE function is only available for users with an extended Prode license.

$model := 1$

$stream := 1$

$diam := \frac{1\ \text{in}}{\text{m}} = 0.0254$

$rough := .00045$

$length := \frac{100\ \text{km}}{\text{m}} = 1 \cdot 10^{5}$

$dHeight := 0$

$dHeat := 0$

$\text{mc\_PIPE}(stream, model, diam, rough, length, dHeight, dHeat) = 0$

The result above will be 0 if the user has a Basic Prode license or 1 for an Extended license.The pressure and phase changes are made in the stream databank.

Parameters :
stream (int) inlet stream
model (int) model for fluid flow and phase equilibria (see below)
diam (double) pipe internal diameter
rough (double) parameter defining relative pipe roughness
length (double) length of this segment
dHeight (double) height difference (inlet, outlet)
dHeat (double) heat added, removed
Codes for models
1Beggs & Brill / Hazen-Williams / AGA
additional models on request to Prode

File save

`mc_AFSave("C:\ProgramData\prode\test.ppp") = 1`   save modifications to a new archive