Free Fluid Manual

22nd March 2021

Contents

Part I.

Introduction and usage

1. Introduction

The FreeFluids soft has been developed along years as an aid for personal chemical engineering calculations, and is now licensed as open source according to the General Public License version 3. It is formed for five different applications. The first and basic application is a physical properties calculation module called FreeFluidsC. It uses correlations, EOS, and activity models, for the calculation of physical properties and equilibrium between phases and, as its name indicates, it has been developed in C. The second module is a GUI for the calculation module, called FreeFluidsGui, that allows to do the most normal calculations needed for pure substances and mixtures, and has been developed in C++ using QT 5.

In order to perform the calculations we will need always some data about the substances used: Tc, Pc, coefficients for correlations..., so a way of storing and retrieving this information is needed, and this is the third module: a database that has been implemented in MS Access (with a graphical interface for its consultation and update). And in SqLite without the dedicated graphical interface, although we can manipulate the database with any of the existing tools for SqLite, and FreeFluidsGui gives also some access to it.

The direct interactive use of the calculation module is interesting, but its real power is developed when we automatize the access to it from engineering packages that are normally strongly needed of physical properties calculations. The interface with Modelica is being implemented now in the module FreeFluidsModelica. At the moment it contains four Modelica.Media compatible packages: IdealGasMedia, IdealGasMixture, TMedia and ExternalMedia. The ideal gas packages are similar to the NASA ones in the Modelica library, but can use many types of correlations for the ideal gas heat capacity at constant pressure. The third one extends Modelica.Media.Interfaces.PartialTwoPhaseMedium, and covers single substance liquid and gas phases, with fast calculations based on correlations. The ExternalMedia package, based in equations of state (EOS), gives high quality thermodynamic properties for single substance, multiphase mediums. Some more packages will be developed in the future, mainly for mixtures.

The engineering module contains some packages for fluid flow calculations: Interfaces, Valves, Pipes and Pumps. The modules for heat transfer, and distillations have not been published yet.

2. How to use FreeFluids

There are two basic ways of using FreeFluids, the Modelica part is of course used as a library in a Modelica simulation environment. The first is to use the GUI for obtaining immediate access to the normal calculations. The second, for programmers, is to link against the C routines in FreeFluidsC to obtain full access to the possibilities of the software. But, before it is necessary to speak about how the necessary data will be accessed.

2.1. Substances and mixtures data

For all the calculations we need data regarding the products used, for example the critical temperature and pressure, or other properties, depending on the calculation to be done. The pass of the data to the calculation functions is done in FreeFluids mainly in the form of structures. These structures are in fact containers for data of the products we are using. The used structures are defined in the file 'FF_basic.h'. We can, inside a program, declare such structures, fill them with data and use them for performing calculations. Of course this is impractical, as the data should be gathered once, stored, and re-used when needed. So it is necessary some form of storage of data. In some programs, like in CoolProp, this storage is done in files (one file for each substance normally). As the normal place for storing data seems a database this has been the solution used in FreeFluids. There is a database called 'Substances.mdb' with all the tables needed for data storage and a GUI to interact with them. There is also a second database called 'Substances.db3', with the same information in SqLite format, but without the interface. Although it is easy to use the same interface for it, if we register the 'Substances.db3' database in Windows as an ODBC data source, later we can link the interface with the tables in this data source.

The problem with this approach is the difficulty to connect from the C program to the database for retrieving the information. In C++ it is easy, as Qt already comes with the necessary libraries to connect with ODBC data sources in Windows or, if we have not registered the data source or we are not in Windows, natively with different databases. In C this is not so easy, and the adopted solution has been to use files containing the data. When you are using the GUI, as it is written in C++ with the Qt5 libraries, it retrieves the data from the database, fills the needed structures with the data, and calls the needed function, passing it the structures. Nevertheless, also in the GUI, there is one information that is retrieved always from files, it is the information related with the UNIFAC subgroups and the interaction between groups. As this information needs to be accessed directly by the C program it has been stored in files. These files must be in a subdirectory labelled 'Data' located in the same directory that the main program.

2. How to use FreeFluids

When we want to use the database information in a C program we need previously to export it, using FreeFluidsGui, as a binary file, and place this file in a subdirectory labelled 'Data', just as with UNIFAC data.

The database stores:

1. Basic data for the substances: MW, CAS number, boiling temperature, critical properties etc. All data are in SI units, with the exception of dipole moment (Debye), quadrupole moment(Debye $\cdot$ m), and Van der Waals area and volume (A). The data is stored in the Products table, and each product is identified by an unique Id.

2. Coefficients for the correlation of physical properties. The equations used are defined in the table CorrelationEquations. The combination of equation, and input and output units, identifies completely how the correlation is used. The different combinations are defined in the table Correlations. And in the table CorrelationParam we have the value of the parameters, identifying the substance and the correlation for which these parameters apply.

3. EOS data. The different EOS used are defined in the table EOS. In the table EosParam you will find the parameters for each EOS, referencing the product and EOS for which they apply. Due to the special complexity of the parametric EOS, in this table only the number of terms of each type is indicated for these EOS, and the values for the parameters are stored in the table SWparam.

4. Substance description for different group contribution models: different flavours of UNIFAC, Bondi, Joback. The description is stored in tables that link the substance Id, the group Id and the number of occurrences of the group in the substance. The tables are named Product_Model

5. Binary interaction parameters (BIP) for EOS and activity models. The supported activity models are defined in the table ActivityModels. The BIP are stored in the tables EosInteraction and ActInteraction. There are 12 parameters for each pair, the first 6 are for the pair i,j and the last 6 for the pair j,i. So, if you need to enter the BIP for the mixture acetone-water, you use the first 6 values for the acetone-water pair, and the last 6 for the water-acetone pair. It seems too many parameters, but the number has been selected in order to give compatibility with the Aspen format. Each register in the table identifies the first and second elements of the pair, the EOS type and mixing rule (the model in case of activity) for which they apply, and the formula to use for the computation of the BIP from the parameters. The different formulas are defined in the table InteractEquations. For cubic EOS with VdW or PR mixing rules, the first 3 parameters are for the calculation of k, using the formula $k = a + b * T + \frac{c}{T}$, and the forth for the calculation of l. In this case $k_{i,j} = k_{ji}$ and $l_{i,j} = l_{ji}$. For MKP mixing rule, only the first 3 parameters are used for the calculation of k, but $k_{i,j} \neq k_{ji}$. For SAFT type EOS the first 3 parameters are used to compute the k for the ε_{ij} calculation, using also the formula: $k = a + b * T + \frac{c}{T}$. For activity models it is also necessary to take into account the units used for the energy.

The database has been filled from different sources but only a small part is delivered with the program. Nevertheless you can populate it with data from several interesting projects as: Caleb Bell's Thermo, J.J. Gomera's PychemQt, Chemsep, CoolProp, CHERIC, AIST, GESTIS, ... I have used Python scripts to automate the transfer and probably I will publicate them in the future.

2.2. The GUI

2.2.1. Download and installation

If you are using Windows the only thing you need to do is to download the repository FreeFluidsGui as a zip compressed file, decompress it in your computer and copy the folder «windowsexe» in your preferred place. If you are just going to use the program you can delete all the other folders, that are intended for compiling or modify the program. The program needs no installation and is run just double clicking on the FreeFluidsGui.exe file. If the MS Access database 'Substances.mdb', inside this folder, has been previously registered as an ODBC source with the name 'Substances.dsn', the program will use it. If not, it will look for a database called 'Substances.db3' in its own directory. Of course if you use a registered ODBC source you can put 'Substances.mdb' in the directory you want.

2.2.2. Single substance calculations

2.2.2.1. General, and Substance EOS calc tab

You select the substance to use (from the connected database) at the combo-box located at the top of the program window. The substances are now arranged in alphabetical order, but I expect to add a CAS based alternative. When you select a substance a FF_SubstanceData structure is loaded with its basic data, and two more combo-box are loaded with the options available for this substance. You will find these combo-boxes at the top of the 'Substance EOS calc' tab. These combo-boxes allow you to select the EOS and the ideal gas heat capacity correlation to use. The selection of an EOS is mandatory. If you do not select a Cp0 correlation the energy related properties will not be computed. When you select an EOS or a correlation, its data is charged in the FF_SubstanceData structure.

The EOS implemented are: different flavours of Cubics, PCSAFT including association, polar PCSAFT using the dipole terms of Gross & Vrabeck or that of Jog & Chapman, VR SAFT with Mie potential, and high precision reference equations using the multi-parameter methodology of Schmidt &Wagner.

Once you have selected the EOS and the ideal gas heat capacity correlation, you specify the pressure at which the calculation will be performed and the minimum and maximum temperatures. If you check the label 'Calc.saturation prop.', vapor pressure, saturated densities, and vaporization heat calculations will be added. When you press the button 'Calculate' you will obtain 11 calculations between the minimum and the maximum temperatures. The results of the calculation is displayed on the table; they are mainly obtained by EOS calculation, but at the bottom of the table, after the derivatives of the reduced residual Helmhotz energy, you will find some additional calculations.

First there is the liquid density, calculated according to the correlation selected on the "Correlations/Tools" tab. If no correlation has been selected, no data will be here. In the next row the Rackett equation is used to calculate the liquid density. A pressure correction is applied to both densities, according to the Chueh-Prausnitz method.

It follows the liquid viscosity. As no reliable general equation has been found, it is calculated only if a correlation has previously been selected in the corresponding tab. On the value given by the correlation, the pressure correction is applied using the Lucas method.

For liquid thermal conductivity and surface tension, the selected correlations are applied. Below the correlation results there are the results applying selected prediction methods. The thermal conductivity is calculated according to the Latini methodology, and the surface tension according to Sastri-Rao and to McLeod-Sugden. No pressure correction is applied later.

Gas viscosity and gas thermal conductivity are calculated by the selected correlations, and also by predictive methods. Lucas for viscosity, and Chung for thermal conductivity. Later a pressure correction is applied to all results.

The liquid heat capacity at constant pressure (Cp) is also calculated according to the selected correlation and to the Bondi method.

The boiling point, at the defined pressure, is also calculated and displayed. The values of H and S displayed are always referenced to the ideal gas state at 298.15ºK and 101325 Pa.

In plus to the main calculation from known P and T, you can perform alternative calculations from known thermodynamic variables. On the left of the tab, you select the known variables, fill them with data and press the 'Calculate' button located at the bottom. The unknown variables will be filled.

At the left bottom of the tab you will find some check boxes and a 'Transfer' button. Checking no more than two boxes, and pressing the 'Transfer' button will copy the selected data in the columns 5 and 6 of the 'Substance Correlations/Tools' table. The finality of this will be explained in the next section.

On the top right of the tab you will find two buttons: 'Export calculations' and 'Export substance as structure'. The first one will export the results in the table in 'csv' format, that can be imported for example by Excel for graphic representation. The second will create two files in the user selected directory. The one with extension '.sd' will contain the FF_SubstanceData structure, with all the data charged, in binary format. The file with '.txt' extension will contain the Modelica package definition of the substance, to be copied to the FreeFluids.ExternalMedia.Fluids package.

2.2.2.2. Substance Correlations/Tools tab

It will work with the substance selected in the upper most combo-box. There are three different functionalities: Calculation and display of physical properties, for the selected substance, using the correlations stored in the database. Regression of parameters for correlations and EOS from data entered in the main table. Finding, and storage in the database, of the critical point given by a SAFT type EOS.

For the calculation and display of physical properties from the stored correlations, the use is simple: You select the correlation to use at the combo-box, the minimum and maximum temperature for the calculation, and the initial row, and number of rows to calculate. You press the 'fill table' button and you will obtain the result in SI units. The selected correlation is charged in the FF_SubstanceData structure, so when you export the substance (described in the previous tab) it will bring with it the parameters of the used correlations. The correlations that have been selected (and

are charged) are displayed at the top right of the tab, and you can control if they are displayed or not in the next calculation, by checking or not the check boxes.

At the bottom of the table are buttons for clear the table, exchange columns, and apply common operations to a column (normally for unit conversion).

You can perform also extended corresponding states (ECS) calculations for the transport properties of the saturated liquid phase. You will find a description of how to proceed in the tooltip popup.

The second functionality is the regression of parameters from data. Two type of parameters can be regressed: parameters for physical properties correlations, and parameters for EOS. Lets review the first type: You need to put the data on the main table, with the first column being always for temperature (or more generally for the independent variable). On the second column you put the data of the physical variable (dependent variable). You can fill these data manually, import from a file with ';' delimited data, of fill with a previous calculation. Once you have the data in place, tell the system from which row to which row you have filled the data, select the correlation to use for the regression and define limits and initial values for the parameters to find. This last pass is normally not necessary as the system will use normal limits, if you do not specify any. If you specify a limit for a parameter you must supply low and high limits and initial value. Once you press the 'Find corr.coef.' button the result for the best parameters will be displayed at the bottom of the tab in 30 seconds. For some calculations it is mandatory to fill previously (If it has not been filled automatically, or you want to use different values) some data, for example critical temperature and pressure for Wagner equation. Some of the correlations are not explicit functions of T, by swapping the columns of temperature and vapor pressure you can regress the correlation for boiling point. And by bringing the enthalpy, obtained from a Cp correlation, to the first column, and the temperature to the second, you can find the coefficients for the correlation of T as function of H. Finally by transferring the liquid density and the logarithm of the reduced bulk modulus, from the EOS calculation tab to the correlations/tools tab and bringing the density to the first column and the bulk modulus to the second, you can regress the correlation for the liquid reduced bulk modulus, and add it to the database.

Once you have the result of the parameters you can add them to the data base. This is done at the right bottom of the tab by pressing the 'Add corr. to DB' button. Previously you must specify which correlation are you adding (not all the correlations can be added), the valid limits for the data used, and any other comment.

For the EOS parameter regression, there are two possible situations: you are regressing for a cubic EOS, using the critical values displayed at the top of the tab. In this case the regression take into account only vapor pressure data, located at the second column of the table. This is the case for almost all cubic EOS variations, except for 'PR 78 fit Tc,Pc,w' and for 'PR fit b,a,w and Tc'. In this two cases density data (kg/m3) located on the second column is also used. For SAFT type EOS you must fill vapor pressure and density data, but the format is more flexible than for cubic EOS. As not necessarily you will have, at a defined temperature, data for both vapor pressure and density, you can enter rows with only vapor pressure or only density data. For the rows with density data it is mandatory to specify the pressure at which the data is obtained, if it is at saturation pressure you enter this pressure on the second column. If the pressure doesn't correspond to a saturation pressure, you enter this pressure on the forth column. Always in Pascal. Before to proceed with the calculation you indicate the range of rows to use, the maximum time to use in the optimization in minutes (if you left to 0 it will take 30 seconds), and the number of dipoles in the molecule (only if you are using a polar SAFT equation).

In order to speed up calculations, when density data is used, a screening system is used for the possible solutions. You specify the maximum error

acceptable for the critical point (Zc) and for the the liquid density. All solutions over these limits will be automatically rejected. If you are regressing data far from the critical point you can go up with the Zc filter (20% error could be OK), but if you are regressing data close to the critical point is better to maintain it lower (7-8% could be OK). The obtained parameters and the errors are displayed at the bottom of the tab.

As with the correlations you can add the result of the optimization to the database. In this case you do not need to specify the type of EOS (it is taken automatically from the regression), but you can add limits and comments.

After the regression of an EOS the parameters are automatically charged to the FF_SubstanceData structure and you can return to the 'Substance Eos calc.' tab and perform the calculations with the parameters found. And transfer later the result of the calculations to the 'Substance Correlations/Tools' tab to compare with the regressed data.

The selected substance, with the correlations previously selected, can be exported in a format adequate for the FreeFluids.MediaCommon.DataRecord used in the Modelica medium definition of FreeFluids.IdealGasMedia, FreeFluids.IdealGasMixture, and FreeFluids.TMedia. You need to export the substance, better with .txt extension, edit it with a text editor, and copy to the FreeFluids.MediaCommon.MediaData package. Later you can use this data in the medium definition extended from IdealGasMedium or from TMedium.

The last functionality is the calculation and storage of the critical point given by a SAFT type EOS. You must select the substance and load the SAFT type EOS to use. If you have just calculated the coefficients for the SAFT EOS, it is already loaded. You have two buttons at the bottom of the form: one for performing the calculation, and the other for storing the result in the database.

2.2.3. Mixture calculations

2.2.3.1. Mixture Calc. tab

You define the mixture to use, and the conditions, in this tab, and you obtain the results in the 'Mixture Results' tab.

You need to begin by selecting the substances that will form part of the mixture. This is done by selecting them in the upper combo-box and by pressing the button 'Add subs.' The substance will be added to the table on the upper left. If you are going to do a calculation that need a fixed composition, you need enter the quantities in mass or moles and when pressing the 'Calc. fractions' button they will be passed to molar and mass fractions. If you are going to calculate the full envelope it is not necessary to enter these quantities.

Once specified the composition, it is necessary to establish the thermodynamic model to use. This is done by selecting options from the different combo-boxes. First it is necessary to choose the model for the liquid phase. At the combo-box 'Liquid phase model' you must choose between Activity or Fugacity(EOS). Next you must choose the EOS for the gas phase (it will be used also for the liquid if you have chosen Fugacity). Once you have selected the EOS you must select the exact EOS you are going to use for each substance. And depending on the properties you want to calculate you must choose also the Cp0 correlation to use.

Once established the models for the gas and liquid phases, it is still necessary do some selections. Once of them is the selection of the mixing rule to use. If you have selected Activity as the liquid model, you must select 'VdW with out int.param.' if you selected cubic EOS for the gas phase, or nothing if you selected a SAFT EOS for the gas phase. Otherwise you select the mixing rule that will be used both for the liquid and gas phase.

If you selected Activity for the liquid phase, or you selected a cubic EOS with a gE mixing rule, you must select now the exact activity model to use (UNIQUAC, UNIFAC ....). Otherwise this step is not necessary. If Activity has been selected for the liquid phase, you must also choose if the reference fugacity will be the vapor pressure of the pure substances, or its fugacities at the same conditions. But I have observed that calculations using Vp as reference are not performing.

Finally, if necessary, you must define the binary interaction parameters. By default they are initialized to 0 and are not necessary for UNIFAC activity models. If you are using an activity model not based on UNIFAC the interaction parameters will be used for this model, otherwise they will be used for the selected EOS. The system of entering the parameters (if needed) is the following: You touch on the left table 'Pair substances selection' the position of the pair to specify. If there are values in the database for this pair you will be able to open the combo-box 'Binary interaction parameter selection'. You select the option you want and the values will be charged on the table ready for use. Otherwise you can enter the values manually for the selected pair.

When you finish all the selections you press the 'Create system' button and the system will be ready for performing calculations. After creating the system you can still change the quantities in the composition and the thermodynamic variables at which the calculations are done. But if you need to change the substances used or the thermodynamic models you will need to press again the 'Create system' button.

When the system has been created you can export it, if needed, to a file. A FF_MixData structure, containing the thermodynamic model selected, and a array with the composition will be exported in binary format. You can later import this file to charge the thermodynamic model selection and the composition, in order to avoid duplicate work entering data.

Once you have created or imported the system you can perform the following calculations:

1. Stability check: According to the Michel L. Michelsen methodology or with a global optimizer, depending on your selection on the radio buttons. If the modified tangent plane distance is lower than $-1e^{-4}$it indicates a unstable composition. It should be just when it is below 0, but probably due to numerical precision problems you can get tpd of for example $-1e^{-6}$ for stable systems.

2. P,T flash calculation: Again according to the Michelsen methodology, or with a global optimizer (improved simulated annealing, or differential evolution). All methodologies will manage VL and LL flashes, but for LL the global optimizers are probably more adequate.

3. Bubble and Dew points calculation for a definite composition.

4. Complete composition-temperature, or composition-pressure phase envelopes.

2.2.3.2. Mixture results tab

In this tab you will find the results of the flash and phase envelope calculations. In order to confirm that the result of the flash has been stable phases, you can check its stability. Select the phase you want to check and press the check button. The algorithm used for the check will be the one selected in the Mixture Calc. tab. Any value lower than $-1e^{-4}$will indicate an unstable phase.

You can perform also the calculation of the transport properties of the selected phase. You must indicate if you want the phase to be treated as liquid or gas.

2.3. Modelica Media packages

A Modelica package called FreeFluids will cover both the media and the engineering calculations. The subpackage MediaCommon contains the general data needed by the other media packages, mainly the individual substance data, according to the DataRecord record. The record for each substance can be filled manually, but the best way is to export the substance from the database, using the FreeFluidsGui. The exportation file can by placed where you want; later you open the file with a text editor and copy the part of the content needed in the FreeFluids.MediaCommon.MediaData package.

Finally, in order to use the substance, it is necessary to extend the package that covers the calculation you need, giving it the substance name and assigning the origin of the data.

On FreeFluids.TMedia.Tests (Test1A and Test1B) you can see the performance of the TMedia package against the WaterIF97_ph standard package of Modelica. and in FreeFluids.ExternalMedia.Tests you can compare the performance of different EOS.

Part II.

Packages description

3. FreeFluidsC

To be filled in the future.

4. FreeFluidsGui

To be filled in the future.

5. FreeFluidsModelica

5.1. MediaCommon package

Provides the general data and functions that are common to several media packages

5.1.1. DataRecord and associated packages

The DataRecord record in the MediaCommon package defines the container for the individual substance data. This data will be used later by the media packages written directly in Modelica language: IdealGasMedia, IdealGasMixture and TMedia. It contains basic data regarding the substance: name, description, CAS, chemical family,...and correlations for several physical properties, normally as function of temperature. Each correlation has: equation to use for its calculation, coefficients, and limits of usage.

The data for each individual fluid is inside the subpackages: MediaDataAL, and MediaDataMZ. Each data is defined as a constant DataRecord, with the name of the substance.

You can create a new record, in those subpackages, copying the MediaDataTemplate (it is inside the MediaCommon package) and filling it manually. Nevertheless the faster and more convenient way is to create the record from the FreeFluidsGui program. You need to select the substance from the database, select the correlations you want to be included in the record, and export it in Modelica format. You can put the file in any place, better with the .txt extension. Later you edit the file, copy its content, and paste it inside the MediaData package. You still need to declare the substance inside the Media packages, filling the name and the origin of the data, that will be the record you just copied.

When using the data in the IdealGasMedia package, only the Cp0 is needed, but you can use also gas viscosity, and gas thermal conductivity correlations. The vapor pressure correlation is also recommended, in order to check if we are working in the gas state.

When using the data in the TMedia package, you will need at least the following correlations: saturated density, vapor pressure, liquid heat capacity. Plus vaporization enthalpy and saturated gas density, if you want to work also with the gas phase. Transport properties correlations as for your needs, and reduced liquid bulk modulus if you want to work at high pressures (till 200 bars). If you want to force the liquid state you can set Tc at a high value, and Pc at a low value (look as example to the MarlothermSH medium).

5.1.2. Functions package

Contains the functions that are common to several media packages. It follows their description.

- CorrelationSolver: It uses the Anderson-Bjork modification of Regula Falsi method for the inverse solving of a correlation. The correlation (f) is defined as a replaceable function. You must extend the CorrelationSolver redeclaring f as the function you want to use. This function must return a Real value, and must accept the calling parameters that will be used by the solver.
- PhysPropCorr: Provides the calculations of physical properties as function of one independent variable (normally T). It needs as input: the number of the correlation to use, the coefficients for the correlation, the molecular weight of the substance, and the value of the independent variable. It returns the value of the physical property. It has an equivalent written in C inside the FFphysprop.c file in the Resources directory.
- PhysPropCorrInv: Performs the inverse solving of the PhysPropCorr function.
- SpecificEnthalpyCorr: Provides the calculation of the specific enthalpy as function of T. You can use correlations using ideal gas constant pressure heat capacity, or liquid heat capacity. Inputs are the same and it has also its equivalent in C.
- SpecificEnthalpyCorrInv: Provides the calculation of T as function of enthalpy. It needs as input: the number of the correlation to use, the coefficients for the correlation, the molecular weight of the substance, the maximum value for the temperature and the value of the enthalpy. It returns the value of T.
- SpecificEntropyCorr: Provides the calculation of the specific entropy as function of T, in similar way than enthalpy.
- SpecificEntropyCorrInv: Similar to SpecificEnthalpyCorrInv but for entropy.
- SpecificEnthalpyCorr2: Receives a DataRecord record as input instead of correlation parameters. Calls the SpecificEnthalpyCorr to performs the calculation of the ideal gas specific enthalpy.
- SpecificEntropyCorr2: Similar to SpecificEnthalpyCorr2 for ideal gas specific entropy.
- Cp0Corr: Calculates ideal gas heat capacity from a given DataRecord and a given temperature.
- Several functions for transport properties calculation.

5.1.3. Types package

Contains definitions not used yet, mainly due to the lack of graphical support for using choices inside mediums in OpenModelica.

5.2. IdealGasMedia package

The medium is designed for single substances in gas phase, at a pressure low enough and a temperature high enough to allow for the ideal gas equation to be used. It extends the Modelica PartialPureSubstance medium. The definition is similar to that of the Modelica.Media.IdealGases.Common. SingleGasNasa, but uses several equations for the Cp0 correlation, as the NASA Glenn coefficients are not available for many organic compounds. Look at the MediaData package information for details on how to use the database to create new substances. The use of a single equation for Cp0 limits somewhat the temperature range. The DIPPR107 equation in J/(kg · K) is recommended

It checks, if the vapour pressure correlation is supplied, that the media is in the gas state, warning if not.

Density is calculated using the ideal gas equation of state. Enthalpy and entropy are calculated from the ideal gas constant pressure heat capacity Cp0, using specific temperature correlations. No constant is added to the raw calculation.

For transport properties, correlations between temperature and the (low pressure) property are used if the constant useTransportCorr==true, and they are available. If not, the Chung method is used for their estimation.

The thermodynamic record contains: p,T,d and h.

As a resume: The medium is for fast calculation of gas phase at low pressure and not too low temperature.

5.3. IdealGasMixture package

The medium is designed for a mixture of substances in the gas phase, at a pressure low enough to allow for the ideal gas equation to be used. It extends the Modelica PartialMixtureMedium. The definition is similar to that of the Modelica.Media.IdealGases.Common.MixtureGasNasa, but uses several equations for the Cp0 correlation, as the NASA Glenn coefficients are not available for many organic compounds. The use of a single equation for Cp0 limits somewhat the temperature range.

It shares the substances data with the other medium models. Look at the MediaData package information for details on how to use the database to create new substances.

It uses also correlations for gas viscosity and thermal conductivity.

Density is calculated using the ideal gas equation of state. Enthalpy and entropy are calculated from the ideal gas constant pressure heat capacity Cp0, using specific temperature correlations.

For transport properties, correlations between temperature and the (low pressure) property are used if available. If not, the Chung method is used for their estimation. Later mixing rules are applied.

The thermodynamic record contains: X,p,T,d and h.

As a resume: The medium is for fast calculation of gas phase at low pressure and not too low temperature.

5.4. TMedia package

There are several very reliable medium packages based en multiparameter equations of state; there are also very simple medium packages based in the ideal gas equation for gases. But for many products we do not have multiparameter EOS, or we do not need their complexity. The TMedia package is based in correlations an offers a good accuracy for liquids till say 200 bars pressure and for gases till 20-30 bars.

The medium is designed for liquid, liquid/vapor, or gas phases. For liquid or biphasic states, its application is limited to the higher temperature limits of the liquid heat capacity and vaporization enthalpy correlations used, but temperature should be lower than 0.85 Tc, and pressure not higher than 200 bars, because at higher values the influence of pressure on properties becomes very difficult to correct for. For gas state, it is limited to the maximum pressure which saturation temperature is below the maximum temperature limit of the liquid heat capacity correlation, but should be limited to a maximum of 20-30 bars. It can't work with supercritical states. It extends the Modelica PartialTwoPhaseMedium and is somewhat similar to the Modelica TableBased medium, but uses specific correlations for each physical property, allows to work with gas phase, and adds a density dependent correlation for the reduced bulk modulus of the liquid, that improves a lot the calculation, at high pressure, of liquid density, isothermal compressibility, and isobaric expansion coefficient. Improving also the calculation of liquid heat capacity at constant volume (Cv) and the speed of sound. The medium properties are obtained using correlations that are mainly functions of T, but different pressure corrections are also used. It uses the substances data stored in the MediaCommon package.

The use of pressure correction is controlled by the constant Boolean 'highPressure'. Its default value is false. If switched to true, pressure correction will be applied (in plus than to liquid density) to liquid specific enthalpy, specific entropy, heat capacity, viscosity and thermal conductivity. It is interesting to make highPressure=true if we need to work over 20 or 30 bars, but the price is a slower simulation.

The values of enthalpy and entropy are calculated from a reference state. The reference state to use can be selected giving value to the constant string 'refState'. The values can be: 'ASHRAE', 'IIR', 'NBP' or 'User'. Any other value will eliminate any correction for the reference state. When using 'User', the raw values at reference_T will be used as zero for both enthalpy and entropy.

The thermodynamic record contains: p,T,gas fraction, d and h. Care must be taken in limiting the use to the temperature limits of the correlations used, as only few checks are done by the media, in order not to interfere with the solver process.

A constant string 'localInputChoice' has been added to the BaseProperties model in order to specify the independent variables to use in each instance of the BaseProperties model. The default value for this constant is the value given to the constant string 'inputChoice' at package level. The valid alternatives are: 'ph', 'pT', 'dT'.

In the package Tests there is a comparison between the medium performance with water and the Modelica WaterIF97_ph medium model (TestA1A/B and TestB1A/B). And with R134A and the Modelica R134a_ph model. The Modelica R134a_ph model seems incorrect. There is also an example taken from ThermoPower (you will need to load the ThermoPower package).

The global idea has been not to use the Modelica files for the storage of substances data, but to store the data in a database, from which we can recover and use them when needed. A database is provided with more than 500 substances that can be enlarged, and the FreeFluids GUI can retrieve

the data from the data base, treat it as needed (for example creating EOS from saturated vapor pressure and/or densities, or creating correlations from the EOS), store the results in the database, and export the data in Modelica format when needed. Nevertheless, in order to make life easier for users, some common substances have been exported, and their packages included in the TMedia.Fluids package.

As a resume: The medium is for fast calculation of liquid phase, condensation, evaporation, and gas phase, when the liquid heat capacity correlation is available. In the liquid and saturated phases, the results are quite good. In the gas phase, the results are better than the ideal gas approach in density and enthalpy. The medium is compatible with OpenModelica 1.16 old and new frontends. The medium is also compatible, since the modification of the BaseProperties model and addition of derivative functions calculation, with Modelica.Fluid and ThermoPower libraries.

5.4.1. Liquid phase properties

The saturated density is calculated using a dedicated correlation. This density is corrected for pressure influence. If the coefficients for the reduced bulk modulus calculation, as function of density, are available, the correction is done using a very accurate Tait like equation. In other case, a substance specific isothermal compressibility factor (with a default value of 6.667 e-10) is used. The parameters for the reduced bulk modulus correlation (with liquid density as independent variable) are normally not available, but can be calculated from a good equation of state of the multiparameter or SAFT types. This can be done easily with the FreeFluids GUI: you make the calculation with the EOS, transfer the results (density and the natural logarithm of the reduced bulk modulus) to the correlations tab, make the regression of the coefficients, and store the result in the database. It is good to calculate the reduced bulk modulus at a pressure close to 50 bars but, if necessary in order to have liquid phase at the temperature of interest, it can be done at higher pressure. Check that all density data used correspond to the liquid state.

A dedicated correlation is used also for the saturated heat capacity. The use of the saturated liquid Cp correlation, instead of ideal Cp correlation plus vaporization enthalpy, makes possible the use of the medium with substances for which we do not have Cp0 data, and improves the liquid phase thermal properties calculation. This correlation can be also a problem, as many times we only find it with a temperature limit of the normal boiling point. This can be solved using a Cp correlation constructed from a good EOS, using the FreeFluids GUI. It is important not to use data too close to the Tc for the regression (use data till 10 K below the Tc). The best equation for the regression of the liquid heat capacity is the PPDS15 equation. Do not use the ChemSep equations as they are not integrated by the medium to obtain enthalpy or entropy. If highPressure has been made equal to true, a density dependent correction is applied to the Cp.

The liquid enthalpy is calculated from the liquid Cp correlation at saturation. If highPressure has been made equal to true, PV correction is applied. The correction interpolates linearly from full correction below 0.45Tc to none at 0.85Tc.

When going back from liquid enthalpy, or entropy, to temperature, we have two ways: One of them is to fit a correlation that makes this calculation, that can again be made with FreeFluidsGui. The second, that will be used if we left the value of lTfromHsatCorr=0, will calculate in situ the temperature using a solving function, as is done in the IdealGasMedia package.

For water, viscosity is calculated, independent of the phase, using the reference equation as function of temperature and density. For other substances, the saturated liquid viscosity is calculated using a dedicated temperature dependent correlation. Pressure correction, according to Lucas, is applied

if highPressure is set to true.

Saturated thermal conductivity is calculated using the corresponding correlation, if available. Otherwise the Latini method is used. If highPressure is set to true, a pressure(in fact density) correction is used.

Surface tension is calculated also by correlation, or using the Sastri-Rao method, if the first is not available.

5.4.2. Gas phase properties

Saturated gas density is calculated using a dedicated correlation. At temperature higher than saturation, for a given pressure, a temperature correction is introduced.

Enthalpy and entropy are calculated from the saturated values at the given pressure (obtained from liquid Cp and vaporization enthalpy correlations), by adding the increase from the saturated temperature to the given temperature, calculated according to the ideal gas Cp, but with a pressure and temperature correction.

Viscosity and thermal conductivity are calculated by correlations or, when not available, by the Chung method.

5.4.3. Two phases

Transport properties are not calculated for the two phases situation.

5.4.4. How to work with liquid only media, and other recommendations

The best way is to specify a very low critical pressure, and a high critical temperature. The minimum correlations needed are: liquid heat capacity, vapor pressure and liquid density, but both can supply constant values for these variables. The critical temperature must be equal to the maximum temperature of the liquid heat capacity correlation. the reason is that, due to the fact that we use a very low critical pressure, every time that we ask for the saturation temperature at given pressure, we will obtain the critical temperature. And this temperature will be used to get the enthalpy and entropy.

For all mediums it is convenient to declare the critical temperature below the high limit of the vapor pressure correlation, in order to avoid errors if the correlation is used over its higher limit.

5.5. ExternalMedia package

The medium is designed for pure or pseudo-pure substances in liquid, gas, or two phases. The thermodynamic calculations are performed using different equations of state (EOS), implemented in C language, that are called using external object and functions. The C code and the substances data are in the Resources folder.

For each substance there is the possibility of choosing between three EOS types: multiparameter, PCSAFT, or cubic. The external object contains the substance data, and is a C structure, that is exported from the database using the FreeFluids GUI software.

The quality of the results is very high when multiparameter (Seltzmann and Wagner, SW) EOS are available. If not, PCSAFT or different flavours of cubic EOS can be used.

The transport properties are normally computed from temperature dependent correlations, with pressure correction. Nevertheless, for viscosity, the system will perform phase independent viscosity calculation (from temperature and density) for selected substances, using dedicated calculation or extended correspondent states with NIST correction polynomia, when available. In order to use the phase independent viscosity calculation the thermoModel parameter must be 3 (multiparameter EOS), as it is the only way to be sure that the supplied density is reliable. For thermal conductivity the same type of calculations are already programmed, but have been inactivated till finishing testing.

The medium implements all the requirements of Modelica.Media.PartialTwoPhaseMedium. It is compatible with the old frontend of OpenModelica, but not with the new frontend, as it does not support external functions yet. It is also compatible, at least partially, with the ThermoPower library. Compared with other high quality libraries, it is very easy to use and, although it is not as complete as for example CoolProp, or RefProp, the quality of the thermodynamic results is practically the same. In plus there is the possibility of using PCSAFT and cubic EOS for substances for which no multiparameter EOS are available.

The main limitation of the medium, when using SW or PCSAFT EOS, is in the vicinity of the critical point, as no special technique, as for example splines, is used. Nevertheless you can go normally quite close to the critical point with SW, not so with PCSAFT as there is still some problems with its density solver. The medium is much slower than the TMedia one, but it is the price for the better precision and wider range of use.

The ThermodynamicState record is quite big, in order to allow all the thermodynamic calculations with just one call to external functions.

The BaseProperties model has been implemented using algorithms instead of equations. I do not know if this is correct, but seems the logical decision when you know the order in which calculations must be performed.

When extending the medium, the following configuration must be done:

The thermoModel Integer constant must be made equal to 1 for using the cubic EOS, 2 for the PCSAFT, or 3 for the SW.

The referenceState Integer constant must be made equal to 1 for using ASHRAE reference state, to 2 for using IIR, to 3 for using NBP, to 4 for using user reference_T and reference_p. Any other number will produce an unreferenced calculation for enthalpy and entropy.

If you are going to use the BaseProperties model, you must make the inputChoice constant String equal to 'ph', 'pT' or 'dT', as needed. When instantiating the model, you can still change the selection just for the object, specifying the value for the localInputChoice constant.

With the Test1aXXX models of the Tests package, you can compare the performance of the cubic, PCSAFT, IAPWS95 and GERG2004 EOS, against the standard Modelica.Media.Water.StandardWater. Test1bXXX compare R410A, Test1cXXX compare propane, and Test1dXXX compare ethanol.

5.5.1. The C code

The C code is placed in the Resources folder.

The FFbasic.h file contains the basic definition of structures and enumerations used in the code.

The FFmodelicaMedium.c file contains the interface between Modelica and C. Basically the constructor and destructor of the structure that will contain the data of the selected substance. The reference to this structure is passed to the Modelica code, that uses this reference when calling the external functions written in C.

The FFeosPure.c and FFphysprop.c files contain the code that will perform the calculation in C.

5.5.2. The Medium data

We need to define the mediums to be used. This definition is inside the Fluids subpackage. Each definition is very short, just with the minimum information necessary. The most important of it is the mediumName, as this name is the name of the file (with extension .sd) placed in the Resources folder that will be used for charging the data to the C structure.

For each medium you need both a medium definition in Modelica, and a binary file with the data in the Resources folder. Both can be made using the program FreeFluidsGui.exe that has been also placed in the Resources folder. It will access the data base, allow you to select the substance, with the EOS and correlations to be used, and later export it to a binary file, with the data as C structure, and to a text file, with the data to add to the FreeFluids.ExternalMedia.Fluids package.

The program allows also the exportation of the correlations in the format used by the TMedia package.

5.5.3. Transport properties by dedicated calculation or ECS

The calculation of the substance viscosity is managed by the FF_Viscosity function in the FFphysprop.c file. If you have selected to work with the multiparameter EOS (thermoModel=3), that grants a good density calculation, it will check if there is a dedicated calculation from temperature and density (available only for few substances). If not, it will check if the correlation defined for gas viscosity calculation is the 112 (NIST coefficients

for viscosity calculation using ECS) and that the data charged for the reference substance correspond to the needed one. If this is OK the ECS calculation will be applied for viscosity, otherwise temperature dependent correlations, with pressure correction, will be used.

If the EOS is of the cubic or PCSAFT type, correlations will be used if available. If not, the Lucas approximation will be used for gas. For liquid phase, if there is no correlation defined, the calculation will be done using ECS from the reference liquid defined.

The definition of the reference liquid is done at medium package level, giving value to the const String refName, which default value is 'Propane'.

5.6. Interfaces package

Its function is to provide the necessary connectors that will support modularity and graphical capability for the fluid flow, and thermal energy transfer, packages.

5.6.1. FluidPort and derived connectors

The FluidPort connector is the base of all the Interfaces package. Its composition is:

- replaceable package Medium = FreeFluids.TMedia.Fluids.Water constrainedby Modelica.Media.Interfaces.PartialMedium;
- SI.AbsolutePressure P(displayUnit = "bar") "Pressure";
- flow SI.MassFlowRate G(displayUnit = "kg/h") "Mass flow";
- replaceable SI.Height Elevation(start = 0) "Port Height";
- replaceable SI.SpecificEnthalpy H "Specific enthalpy";
- replaceable Medium.MassFraction X[Medium.nX] "mass fractions composition";

An explanation is needed regarding why and how each element is defined.

The package Medium is included in the connector mainly because it seems the standard in Modelica.Fluid. It will be useful if the propagation of the Medium package is done, in the future, by the connectors. It is useful also for initialization.

P and G are according to the Modelica implementation of a connector: one potential variable and one flow variable. This aids to grant that the number of equations and the number of variables are the same if the relationship between elements is done only by connections.

The variables Elevation and H don't conform to the Modelica.Fluid way, that uses stream variables. The Elevation could be avoid, as the only information needed inside the elements with connectors is the elevation difference between ports, not the absolute elevation. But its absence would avoid to enter information, based on elevation data at the ends, directly.

The enthalpy is absolutely necessary if we want to pass energy information between connectors. The solution adopted by Modelica 3.0 was the creation of stream variables, but I think that this complicates too much the situation. The alternative is to declare them as causal (input or output). This seems to prevent from allowing reverse flow, but it is not totally true, as can be seen in several examples. They are declared replaceable in order to allow its declaration as causal in derived connectors.

From the FluidPort connector, a FluidPortA as normal input, and FluidPortB as normal output are derived. The difference is the appearance, the default value for G (positive for the normal input, and negative for the normal output), and the causality of the extra variables. Initialization is also provided for the H and P of FluidPortB, based in the default values of the Medium.

5.6.2. TwoFluidPorts partial model

This partial model contains a FluidPortA PortA, and a FluidPortB PortB connectors, and some equations linking the values of both connectors variables. The more fundamental equation is:

0=PortA.G+PortB.G. This grants that all massic flow entering one port goes out by the other. G is considered positive if the flow enters the port.

Some parameters will configure more possible links between the ports variables:

- parameter Boolean useElevDifference=true will activate the use of an extra equation for ports elevation.
- parameter FreeFluids.Types.ElevationOption elevCalcMethod. If useElevDifference=true, selects the equation to use for the elevation calculation. If the selection is “differential” the equation PortB.elevation=PortA.Elevation+elevDifference is applied. If the selection is “absolute” the equation PortB.Elevation= portBelevation is applied. Being elevDifference and portBelevation parameters with default value of 0.
- parameter Boolean calcEnthalpyDifference=true will control if a calculation of the difference of enthalpy between ports is applied or not. But the calculation to apply remains undefined.
- parameter Boolean passComposition = true will control if the composition of both ports is made equal or not.

Finally two variables, with the equations to solve them are added: Hdiff = PortB.H-PortA.H, and Pdiff=PortB.P-PortA.P.

With this implementation the model is imbalanced in two way: For the two pairs of potential/flow variables, there are three equations(PortA.G=0, PortB.G=0, PortA.G=-PortB.G), so an equation linking these variables is missing. Normally it will be the calculation of the pressure drop as function of flow. The second point of imbalance is that there is no equation for the calculation of the output PortB.H. The extending models must provide the missing equations.

Although not correct according to the Modelica standard, the model is prepared for the connection of more than one PortB to a PortA, with the idea of making simpler and faster some connections. When doing so we must take into account that, in a connection point elevation, enthalpy, and composition, must be supplied only by one connector. If not, we will have an over specification for the value. The situation is different for elevation than for enthalpy and composition. In a connection point all the ports must have the same elevation, so the only thing that we have to do is to

inhibit the duplicate propagation of the elevation, making useHeightDifference=true only in one of the elements that can supply the elevation value to the connection. For enthalpy and composition, it must be possible to connect elements with different enthalpy or composition output, and this can be done using mixers, that for simplification are coded as no reverse flow. But, if the enthalpy variation inside the elements is only due to elevation changes, we can grant that all connectors have the same enthalpy at the connection point, and solve the problem in the same way that we did with elevation, allowing reverse flow. The same is applicable for composition, if composition is constant.

A second point must be considered: In a connection of more than two connectors, regardless the physical size of the connections could be the same, there is normally a change in velocity, so in kinetic energy and enthalpy. This means that the equal enthalpy solution is only an approximate solution valid when the velocity is low (liquids and gases at not high velocity).

As a resume, we will develop elements allowing for reverse flow, based in a fixed enthalpy and composition at each connection point, and elements without reverse flow capability, valid for situations with different enthalpy, or composition, of the connectors.

5.6.3. Sources, sinks and references

As origin/end of the flow, except for loops, we need at least a source, and at least a sink. There are defined two source models: a general FlowSource, valid for one and two phases Mediums, and a FlowSourceSP valid only for single phase Mediums.

The FlowSource model contains a FluidPortB PortB connector, which pressure and enthalpy are generated in different ways, coming from configurable parameters, that fix temperature, pressure, enthalpy or density, or from Real connectors for temperature and pressure. Remember to give the value at the Real input connectors in SI units. In order to enhance the possibility of giving enthalpy start values at the different elements of the model, the local parameter H receives a calculated value when its value is not directly given.

Flow and elevation can be specified, or not, using parameters (or a Real connector for flow). First you choose if the flow or elevation must be defined for the connector, and later you choose if the external value for flow, coming from the connector, must be accepted. If you choose to define the flow, and you don't accept the external value, the local value for the flow will be used. According to the Modelica standard, this situation is incorrect: we should supply always the value of elevation (as it is an output), and never the value of flow, in order to prevent an unbalanced model (as the value of P has been already fixed). But in order to gain in flexibility, the possibility of an overspecified model has been used, taking into account that if we overspecify the FlowSource, a FlowSink must be underspecified.

The implementation of the Real connectors, for receiving temperature, pressure and flow from external sources, has been implemented according to the Modelica standard way for conditional connectors. The connector receiving the external values is conditional, and is connected to a protected connector which, in case that the conditional connector is deactivated, receives its value from the corresponding parameter.

The composition is always fixed at the values entered in the parameters window. Its default value is the default value of the Medium.

The FlowSourceSP model is a simplified version, without the options of generating pressure and enthalpy values at saturated conditions for the PortB.

The FlowSink model contains a FluidPortA PortA connector, provision for receiving fixed flow, or pressure, from external sources, and the following parameters:

- parameter FreeFluids.Types.BoundaryOption fix= FreeFluids.Types.BoundaryOption.fixFlow. Used to select between fix the flow, the pressure or nothing at the PortA.
- parameter Modelica.SIunits.AbsolutePressure P= 1e5. Value used for PortA.P if fixDP is selected.
- parameter Modelica.SIunits.MassFlowRate G. Value used for PortA.G if fixFlow is selected.

The third alternative is fixNone, that will produce an underspecified connector, that must be matched by an overspecified connector elsewhere.

It is very important to understand that the FlowSource and the FlowSink can act both as source or sink for flow. The only difference between them is that the FlowSource will inject values for enthalpy and elevation, and the FlowSink not. In order to improve convergence, as the start for flow is defined as negative (flow leaves the element) for the FlowSource and positive for the FlowSink, we should provide oposite start values if we are using FlowSource as receiving flow, and FlowSink as an origin for flow.

In order to allow for the specification of pressure and elevation at the connection points, there are several extensions of the FreeReference model. This base model contains a FluidPortB connector and just one equation: PortB.G=0. The extensions allow for the specification of pressure and elevation by parameters. This is against the Modelica standard, because in this way we are connecting several outputs to the same input.

Which is the situation if we want to stick to the Modelica standard? The answer is: do not fix flow at the sources, give flow or pressure at the sinks, make all connections of two or more outputs to the same input using mixers, and do not use pressure or elevation references.

5.7. Valves package

5.7.1. ValveBase partial model

It is an extension of the Interfaces.TwoFluidPorts model. It adds the following parameters to configure a two ports valve:

- parameter FreeFluids.Types.ValveFixOption fix = FreeFluids.Types.ValveFixOption.fixKv, in order to select if we are going to specify a fized Kv, a fixed pressure drop or a fixed flow along the valve.
- And the following parameters: fixedKv, fixedDP and fixedFlow, in order to allow the user specification of the needed value.

It adds also the following variables:

Kv for the valve Kv

Cv for the valve Cv

Medium.ThermodynamicState State, in order to specify the state at which physical properties will be evaluated.

Q for the volumetric flow at State conditions.

T for fluid temperature at State conditions.

Rho for fluid density at State conditions.

5.7.2. ValvePartial partial model

It is the base model for control valves, and extends the ValveBase model. It adds the following parameters for configuration:

- parameter Boolean isCompressibleFlow = false. In order to specify the type of flow to consider. If it is made true the flow will be always considered as adiabatic.
- parameter Boolean isLinear = true. To specify the valve characteristics. If made false, the valve will be considered as isopercentual.
- parameter Boolean useFixedAperture = true. If true, the aperture of the valve will be fixed to the value entered manually at the aperture parameter. Otherwise the aperture will be taken from the RealInput connector Opening, making possible the calculation of the aperture outside of the model.
- parameter Real aperture = 1.0. The fraction of aperture of the valve used if useFixedAperture = true.

The following variables are also defined:

- SI.Area KvFlow. The acting Kv of the valve taking into account its aperture, different from the nominal Kv.
- The conditional connector Modelica.Blocks.Interfaces.RealInput Opening, conditioned to useFixedAperture=false. In order to receive the valve aperture by connexion.
- The protected connector Modelica.Blocks.Interfaces.RealInput Aperture.

The implementation of the Aperture connector is done in the standard way: The connectors Opening and Aperture are connected, but if useFixed-Aperture is made true, the Opening connector disappears, and also the connection. And the equation Aperture=aperture (reading the manual entered value) is activated.

There is also an equation relating the Kv, KvFlow and Aperture, taking into account the selected characteristic of the valve.

And an equation relating flow with KvFlow and differential pressure, if Aperture>0. Otherwise the flow is made equal to 0.

5.7.3. ValveIncompressible model

Extends the ValvePartial model, fixing isCompressibleFlow=false.

It adds the parameter Boolean isIsenthalpicFlow = true, in orther to specify the type of flow to consider. If made equal to false, adiabatic flow is considered.

It defines the State (for physical properties calculation) from the PortA variables.

An equation is added relating the two ports enthalpy, if calcEnthalpyDifference=true. If the flow is isenthalpic, they are the same. if it is not, only the differential height between ports is taken into account.

5.7.4. ValveCompressible model

Extends the ValvePartial model, fixing isCompressibleFlow=true.

It adds two Medium.ThermodynamicState StateA and StateB. In order to retrieve the physical properties at both ports. It defines also variables for temperature, density and velocity at both ports: Ta, RhoA,Va, Tb, RhoB, Vb.

The State variable (used for retrieving physical properties for the pressure drop calculation) is defined at the highest enthalpy of the two ports and at the lowest pressure of the ports (but not lowest than half of the highest pressure).

An equation is added relating the two ports enthalpy calcEnthalpyDifference=true. It will take into account the kinetic energy only if the valve diameter is higher than 0.

5.7.5. CheckValve model

Extends from the ValveBase model. It treats always the flow as incompressible and isenthalpic.

The State variable is defined at PortA pressure and enthalpy.

Pressure drop is calculated if PortA.P>PortB.P, otherwise the flow is made 0.

If calcEnthalpyDifference=true, the enthalpy of both ports is made equal.

5.7.6. SafetyValve model

Allows the calculation of the flow or the area of a safety relief valve or disk. It uses an isentropic calculation between the inlet, supposed with velocity equal to 0, and the orifice. The gas flow can be chocked or not, and all the calculations are done with the physical properties calculated

by the Medium, except the possibility of manual entering of the isentropic coefficient. Correction coefficients are applied for backpressure and for viscosity, according to the API 520 methodology. It is capable for biphasic flow calculation, but the model SafetyValveFlash, that conforms to the API recommenations, is probably more adequate.

Two thermodynamic states are declared at the orifice outlet, both isentropics with the inlet. One of them at the discharge pressure of the valve, the other adjusted to obtain sonic speed at the orifice. If the discharge pressure is below the obtained critical pressure, the flow is taken from the critical calculation, if not from the discharge pressure calculation.

5.7.7. SafetyValveStd model

API 520 and ISO 4126-1 calculation methods are implemented, with the API one as default. It is limited to monophasic flow. Although a Medium is declared, and physical properties can be taken from it (except the isentropic coefficient for gases, that must be always entered by hand), there is the possibility of manual entering of all the physical properties. If you enter all physical properties manually, you can use data for a different fluid than that of the Medium. The Medium will be used just for generating P and H at the PortA, with H being passed to the PortB if the calculation has been activated. As H is not used in the calculation of the flow, if the pressures are OK the calculation should be fine.

5.7.8. SafetyValveFlash

It implements the calculation methodology described at API 520 Annex C section C.2.1, based on isentropic flow. In order to get the flow, an array of 25 pressures, between inlet and outlet pressures, are tested, and the higher flow is taken. Instead of implementing the integration of $\frac{dP}{\rho}$ to obtain the velocity, the relationship $\frac{d(v^2)}{2} = -dH$ has been used.

5.7.9. SafetyValveOmega

It a model for safety valves working with flashing liquids, according to API 520 Annex C sections C.2.2 and C.2.3. The parameter Boolean liquidInlet allows you so select the calculation method to use, if made true it will use the C.2.3 methodology, that works with an inlet 100% in liquid state. Otherwise the C.2.2 method will be used. The parameters allow you to choose between fixing the discharge area or the flow, configure the basic characteristics of the valve, and enter the physical properties or let them to be taken from the Medium.

5.8. Pumps package

5.8.1. PumpBase partial model

Its function is to define the common elements needed for different type of pumps, providing just few equations between them.

Parameters:

- forceSpeed=false. If made true, the pump speed will be made equal to the manual defined parameter, or to the speed received via a connector, depending on the selection made.
- useExternalSpeed=false. If made true, the speed received at the connector is used. Otherwise the manually filled parameter will be used. (the forceSpeed parameter must be true for calculation to apply).
- fixedSpeed. The manually fixed speed request.
- numParallelUnits=1. The number of identical pumps working in parallel.
- directFlow=true. If true, the flow will be from port A to port B. And the reverse if false.

Variables:

- N. for the pump speed. In order to assign it, a conditional RealInput connector 'Speed' connected to a protected RealInput connector 'SpeedIn' are used. I 'useExternalSpeed' is false the 'Speed' connector and its connection disappears, and N is made equal to 'fixedSpeed'.
- Qunit and Qtotal. For the volumetric flow of each individual pump, and of all parallel pumps. They are related by Qtotal = Qunit * numParallelUnits. And Qtotal is related to the mass flow by PortA.G = Qtotal * Rho.
- A ThermodynamicState 'State' is defined in order to gate the physical properties of the fluid. The state is defined at the enthalpy and pressure of PortA, as we do not expect too much change for the physical properties between the ports.
- The variables Rho and T, get the density and the temperature of the State.
- The variables specific energy: 'SE', pump efficiency: 'Efficiency', total absorbed power: 'Wabs'. Two equations are provided here, one comes from the definition of efficiency, and the other relates 'Wabs' with flow, differential pressure and efficiency.
- A pump impulsion height variable 'Dh' that express the differential pressure between ports a liquid height. An equation relating Dh with the differential pressure and density is given.

An equation is missing for finding Efficiency, and it is missing also an equation relating N, Qunit and Dh.

5.8.2. BumpPumpBase partial model

It extends the PumpBase model and introduces parameters for the pump curve definition. They are flow, head, and efficiency, at two points of the curve, plus the head at zero flow.

It defines three variables for: flow, head and efficiency with a nonviscous fluid.

It defines two equations relating the three variables with the pump speed. So, if the pump speed is fixing, the definition of one of the three variables will allow the calculation of the other two.

5.8.3. BumpPump model

Is the model for nonviscous fluids.

Extends the BumpPumpBase model and just makes equal the flow, head, and efficiency, of the pump to those of the nonviscous fluid.

5.8.4. BumpPumpViscous model

Extends the BumpPumpBase model, and introduces the viscosity, that is calculated from the ThermodynamicState. The flow, head and efficiency of the pump are related to those of the nonviscous fluid using correction factors derivated from the fluid viscosity.

5.8.5. PositivePump model

It extends the the PumpBase model. the following parameters are defined:

- n0: The nominal rotational speed
- q0: The nomimal volumetric flow at the nominal rotational speed
- r: The fixed efficiency. The equation Efficiency=r is applied.
- qLeak: The pump expected volumetric leak at pLeak differential pressure.
- pLeak: The differential pressure at which the qLeak is induced.

The variable Qleak is defined, with the equation for its calculation: Qleak= qLeak*Pdiff/pLeak. So the leak is proportional to the differential pressure, in both directions.

Finally the equation Qunit=q0*N/n0-Qleak is applied. You can reverse the flow of the pump by giving a negative value to the rotational speed.

5.9. Pipes package

5.9.1. Mixer models

The mixer models are used for obtaining a mixed flow coming from different enthalpy flows. There are mixers for 2,3 and 4 incoming flows: MixerPH, Mixer3PH and Mixer4PH. Composition and elevation are considered the same at all ports, the user must check that this is fulfilled.

Composition and elevation of the output is made equal to those of PortA.

Equations are added in order to make equal the pressure of all ports.

Equations are added in order to make sumatory of all flows, and of all enthalpy flows equal to 0.

These models can act as fully reversible regarding flow and enthalpy, provided that only one connector is the output.

5.9.2. AbruptAdaptor model

The AbruptAcaptor model computes the pressure and enthalpy change in a frictionless reduction/enlargement.

The pressure assigned to a flow port is that measured at the pipe end, taking into account the velocity of the flow, as this is the pressure needed for the calculation of physical properties like density. When making connections, we assume equal pressure at all connecting ports, without taking into account that, if the streams have different velocities, there will be a change in pressure. This makes the balance inexact. If we want to improve the calculation, taking into account the velocity impact, it is necessary to reference pressure and enthalpy to the same velocity. This is the function of the abrupt adapter. It is a frictionless adapter.

If we want to use the adapter, it is necessary to put one adapter at each end of the pipe, except for the connection of two only pipes of the same diameter. In this case the equal velocity is granted at the connection point.

5.9.3. PdiffSource model

It extends the TwoFluidPorts model and allows for the specification of a pressure difference between the ports, maintaining constant the enthalpy. It uses several parameters in order to specify how the pressure difference is calculated:

- parameter Boolean useFixedDiffP = true. To specify if the pressure drop is constant of function of flow.
- parameter Modelica.SIunits.PressureDifference dP. The pressure reference to apply, as fixed or at reference flow.

- parameter Modelica.SIunits.MassFlowRate refG. If useFixedDiffP = false, the dP will be referenced at this flow rate, and a function applied for other flow rates.
- parameter Boolean isLaminarFlow = false. To configure the function to apply for the pressure difference calculation.

5.9.4. Readers

Are used to get the physical properties at a connection, or physical properties and flow of a stream. They have two FluidPorts that are completely transparents: the properties of both ports are made equal. You can connect them just to one port, where they will report 0 flow, or allow flow to pass through.

5.9.5. PipePhysical model

Contains the description of the shape and wall properties of a pipe. Introduces the following parameters for user input:

- parameter SI.Distance lTube = 0. In order to allow direct user input of individual tube length.
- parameter Integer numTubes = 1. To specify the number of existing identical parallel tubes. It is useful mainly for exchangers.
- parameter SI.Distance di = 0. In order to allow user specification of the diameter for circular pipes.
- parameter SI.Area section = 0, and parameter SI.Distance perimeter = 0. In order to specify perimeter and section for non-circular pipes.
- parameter SI.Distance roughness = 1.5e-005. Pipe internal roughness.
- parameter SI.Distance thickness = 1e-3. Pipe wall thickness.
- parameter SI.Density rhoWall(displayUnit = "kg/m3") = 8000. Wall density, for weight calculation.
- parameter SI.Distance thicknessInsul = 0. Insulation thickness, if applicable.

And the following parameters for configuration of the applicable equations:

- parameter Boolean useTubeLength: is used to make equal, or not, the individual tube length to the parameter lTube. If false, the pipe length remains as unsolved variable. This should allow us to retrieve the individual tube length from the total length and the number of tubes, if we have fixed the total length as for example in coils.
- parameter Boolean fixNumTubes = true. To specify if the number of tubes must be fixed at numTubes.

- parameter Boolean isCircular = true: If true, establish the relation between the perimeter and the diameter of the pipe as PathPerimeter=pi * Di. If parameter Boolean useDiameter = true is also true, the equation Di=di is activated. If the diameter Di is not fixed, we will need an extra equation to solve for it. As the equation: Di = 4 * PathSection / PathPerimeter is always active, we can solve Di, PathSection or PathPerimeter. With this definition is clear that Di is the hydraulic diameter except for a double pipe.
- parameter Boolean useDiameter = true. If true and isCircular is also true, the supplied(parameter) diameter will be used, activating the equation Di=di;
- parameter Boolean useSectionAndPerimeter = false. If isCircular=false, making useSectionAndPerimeter=true will activate the equations that make PathPerimeter and PathSection equal to the given parameters. If none of the options for circular or section and perimeter is activated, we will need two extra equations to solve for diameter, section and perimeter.

A lot of extra variables are calculated for the pipe, related with length, diameters, surfaces, volumes and weight.

5.9.6. Pipe partial model

Is an extension of the PipePhysical and of the TwoFluidPorts models, that gives the basis for flow definition. Introduces the following parameters for user input:

- parameter Integer numActiveTubes=1. Although the physical pipe already has the parameter numTubes, it is possible use only some of them as active for flow.
- parameter Real pipeComplexity = 0. Allows a fast calculation of the equivalent tube length without detailing the existing accessories. Approx. estimation of equivalent length: 0=not used, 0.25=supply lines, 0.5=long runs, 1=normal, 2=valves, 4=complex valves. If it is 0, the equivalent length will be calculated using the equivalent length of accessories.
- parameter Real equivL_Di = 0. Used to give the equivalent length of existing accessories.
- parameter Real numVelocityHeads=0. In order to add pressure losses expressed as number of velocity heads.
- parameter SI.Area kv = 0. Gives the possibility to add Kv of valves and other accessories to the pressure drop calculation. This will be used in plus to the equivalent length used.
- parameter Fraction aperture = 1. Fraction of the Kv that will be applied.

And the following parameters for configuration of the applicable equations:

- parameter Boolean fullBore = true. If true activates the equations: PathSectionActive = PathSection, and PathPerimeterActive = PathPerimeter. If the pipe is double, it will be necessary to add the internal perimeter, and to subtract the internal section. If it is a falling film pipe, the flow section will be reduced. The final hydraulic diameter that will be used is calculated as: Dh = 4 * PathSectionActive / PathPerimeterActive.

- parameter Boolean isCompressibleFlow = false. If true, the velocity change between ports will be used for the momentum and energy conservation equations.

Several variables are calculated: Surface and volume for the active tubes, PathSectionActive, PathPerimeterActive, Dh, Slope.

5.9.7. PipeFlowBase partial model

Is an extension of the Pipe model. It is in fact the base model for all thermal and non thermal pipes, one or two phases.

It adds one important configuration parameter:

- parameter FreeFluids.Types.ThermalType thermalType. If its value is isenthalpic will activate PortA.H=PortB.H. If adiabatic : W = 0. If isothermal: Ta=Tb. If fixedPower: W= fixedW. If fixedDeltaT: Tb=Ta+fixedDeltaT. Otherwise an equation must be defined for the heat transfer calculation.
- parameter Modelica.SIunits.HeatFlowRate fixedW = 0. The exchanged heat if thermalType=fixedPower.
- parameter Modelica.SIunits.HeatFlowRate fixedDeltaT = 0. The differential T to apply if fixedDeltaT has been selected.

It defines Medium.ThermodynamicState records for PortA(StateA) and PortB(StateB) obtained from the values of P,H and X at the ports. From them, the density, velocity, volumetric flow, and temperature are calculated.

It contains the momentum and energy conservation equations, with two missing variables: the pressure drop and the exchanged heat. The last only in the case of a detailed heat transfer model.

5.9.8. PipeSimple1Ph model

Is the model for a one phase flow in a completely filled single pipe, with no detailed heat transfer model, you can use it with the options: isenthalpic, isothermal, adiabatic and fixedPower. It extends PipeFlowBase, defines the average thermodynamic state as that at average pressure and enthalpy between the ports, and implements the calculation of the pressure drop. the flow direction is fully reversible.

If used with thermalType= detailed, the calculation of the heat exchanged (W) must be added to the model.

5.9.9. PipeSimple2Ph model

Is the model for a two phases flow in a completely filled single pipe of a saturated medium, with no detailed heat transfer model. It extends PipeFlowBase, defines the average thermodynamic state as that at average pressure and enthalpy between the ports, and implements the calculation of the pressure drop according to the Muller-Steinhagen and Heck method. It is similar to PipeSimple1Ph model regarding heat exchange.

5.9.10. PipeThermalBase partial model

Extends the PipeFlowBase model and adds the minimal definitions shared by all types of thermal pipes, when a detailed heat transfer model, for calculation of heat exchange, is going to be used. It contains the following parameters for user input:

- parameter SI.ThermalConductivity kWall = 16. The thermal conductivity of the pipe wall
- parameter SI.ThermalConductivity kInsul = 0.04. The thermal conductivity of the insulation.
- parameter SI.ThermalInsulance foulingF = 0.0002. The fouling factor at the internal fluid surface.
- parameter Real emissionCoef = 0.26. Emission coefficient of the external surface, if radiation heat transfer is present.

And the following for model configuration:

- parameter Boolean useWallsResistance = false. If true the equation relating Twall and Tsurf with the thermal conductivity of wall and insulation is taken into account. Otherwise Twall=Tsurf.
- parameter Boolean fullHTperimeter = true. If false, it will be necessary to define the perimeter zone used for heat transfer, for example in half pipes.
- parameter Boolean fullHTlength = true. If false, the tube length used in heat transfer must be specified. It is the case in partially immersed pipes.
- parameter Boolean useThermalConnector = true. Allows or not the transmission of the heat flow from the thermal connector to the internal variable used as heat exchange.

The following variables are also defined:

- Twall and Tsurf: Temperatures at internal and external surfaces.
- LMTD: The mean logarithmic temperature difference between ports and Tsurf.
- H: The heat transfer coefficient, to be calculated by the detailed model used.
- SactiveHT. Is the total surface available for heat transfer. If fullHTperimeter is true, the equation SactiveHT = SiActive is activated. Otherwise it is necessary to calculate it.
- SusedHT. Is the total surface used for heat transfer. If fullHTlength = true, the equation SusedHT = SactiveHT is activated.

A HeatPortB connector is defined. It is used for the transmission of the pipe information to the surrounding, in plus to the pipe surface temperature and heat flow. The heat flow defined in the connector is used internally in the pipe only if the parameter useThermalConnector=true. If it is false the heat flow at the connector will have no effect. The use of the connector will be probably limited, as the connection must be done fixing the pipe external temperature. In the case of heat exchangers, the calculation of W will be done using LMTD or e-NTU methods, with no need of connectors.

The following variables remain unsolved:

Pressure drop regarding flow.

H, Tsurf or Twall, and W, if thermalType=detailed, otherwise only H and Tsurf or Twall. The equation for H must be provided by the extending models.

5.9.11. PipeForcedConvection

It is the pipe for forced convection heat transfer in single phase flow. Extends PipeThermalBase model. It must calculate the pressure drop by friction and define a relationship between the surface temperature and the exchanged heat. In order to perform the pressure loss calculation, the Reynolds number and the friction factor are defined and calculate. The calculation of H (heat exchange coefficient) is also performed. Finally a relationship between the heat exchanged and the surface temperature is established using the LMTD. If the parameter useThermalConnector=true, the model is balanced, as the equation for the connection of W (heat exchanged) will be activated. If not, an equation determining the heat exchanged must be present in the receiving model. This will be the system used for more elaborated models of heat transfer than just consider a fixed surface temperature.

5.9.12. PipeThermalFallingFilm

The flow is incompressible. The pressure loss is exactly the geodesical height loss, so there is no relationship between pressure loss and flow, and the outlet and inlet pressure are identical. The flow will depend mainly on the film thickness and slope of the pipe. This minds that the flow must come always from one of the two connectors of the pipe. Other characteristics and limitations are the same than for the forced convection model.

5.9.13. PipeCondensingBase partial model

Defines the variables needed and perform the calculations for gas condensation inside a pipe. We know that we have two possibilities: supply pressure at both ends of the pipe, or supply pressure at one end and flow at the other. If we supply pressure at both ends, the flow will be made independent of the low pressure side, and will be calculate as the maximum flow that can be condensed by the pipe. If the flow is specified at one side, the PortB pressure will be made equal to the PortA pressure, assuming that the pressure drop equals the velocity and height loss, and the vapor quality at the outlet will be calculate. We will have two possibilities:

a. The flow is not totally condensed. This will be indicated by a vapor quality higher than 0.

b. The flow is totally condensed before the end of the pipe. This will be indicated by a vapor quality below 0. In this case a composite calculation of condensation+ subcooling will be necessary.

5.9.14. PipeThermalCondensing

Extends from PipeCondensingBase and contains just the equation needed as for the selected option, plus the calculation of the total mass of fluid inside the pipe, and the assumption that all exchanged heat has been by condensation. By now there is not the possibility of combine condensation plus subcooling in the same pipe.