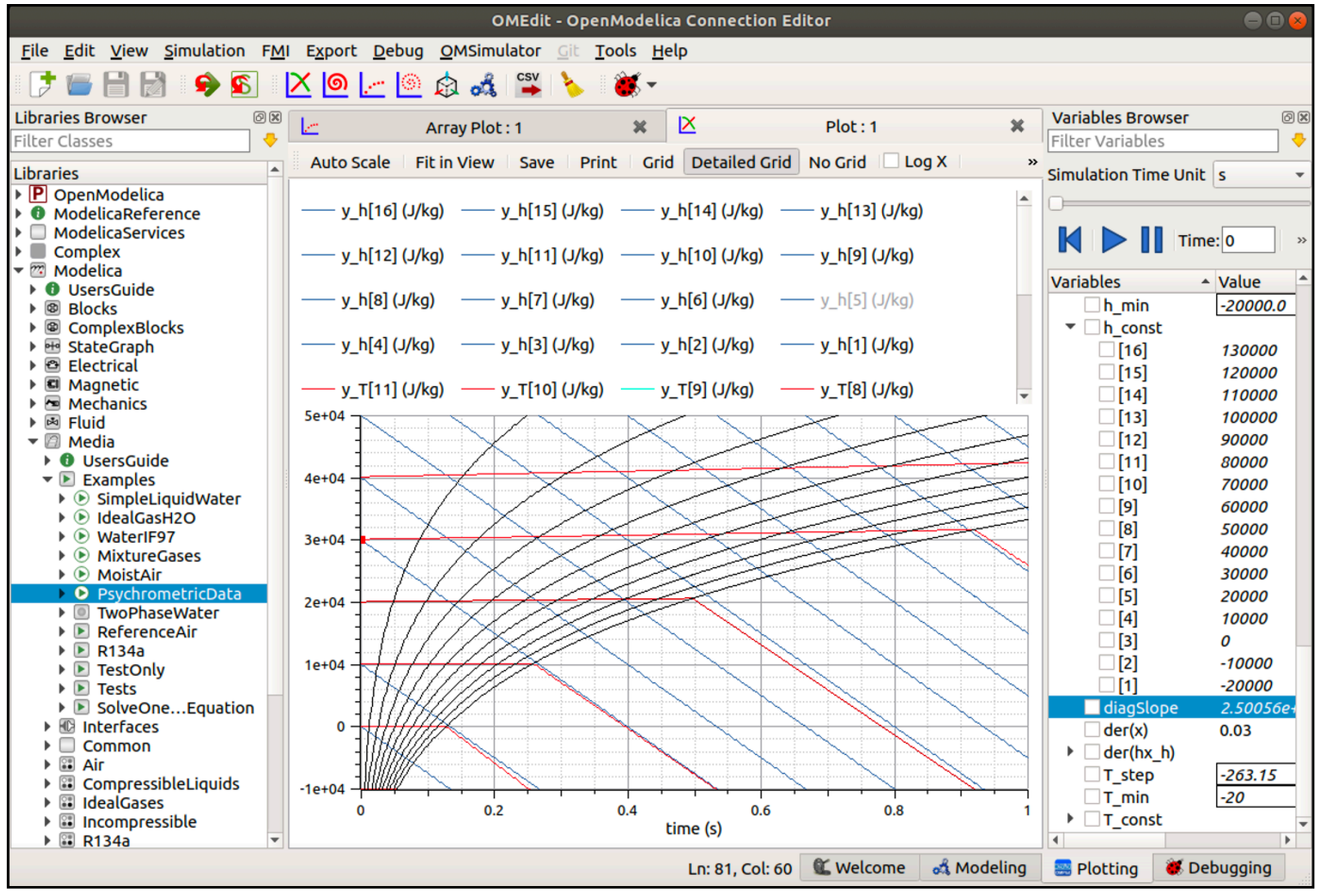
Modelica.Media.Air.MoistAir について



第7回 Modelica ライブラリ勉強会 2019/01/12

第9回 Modelica ライブラリ勉強会 2019/03/23

第10回 Modelica ライブラリ勉強会 2019/04/27

Finback

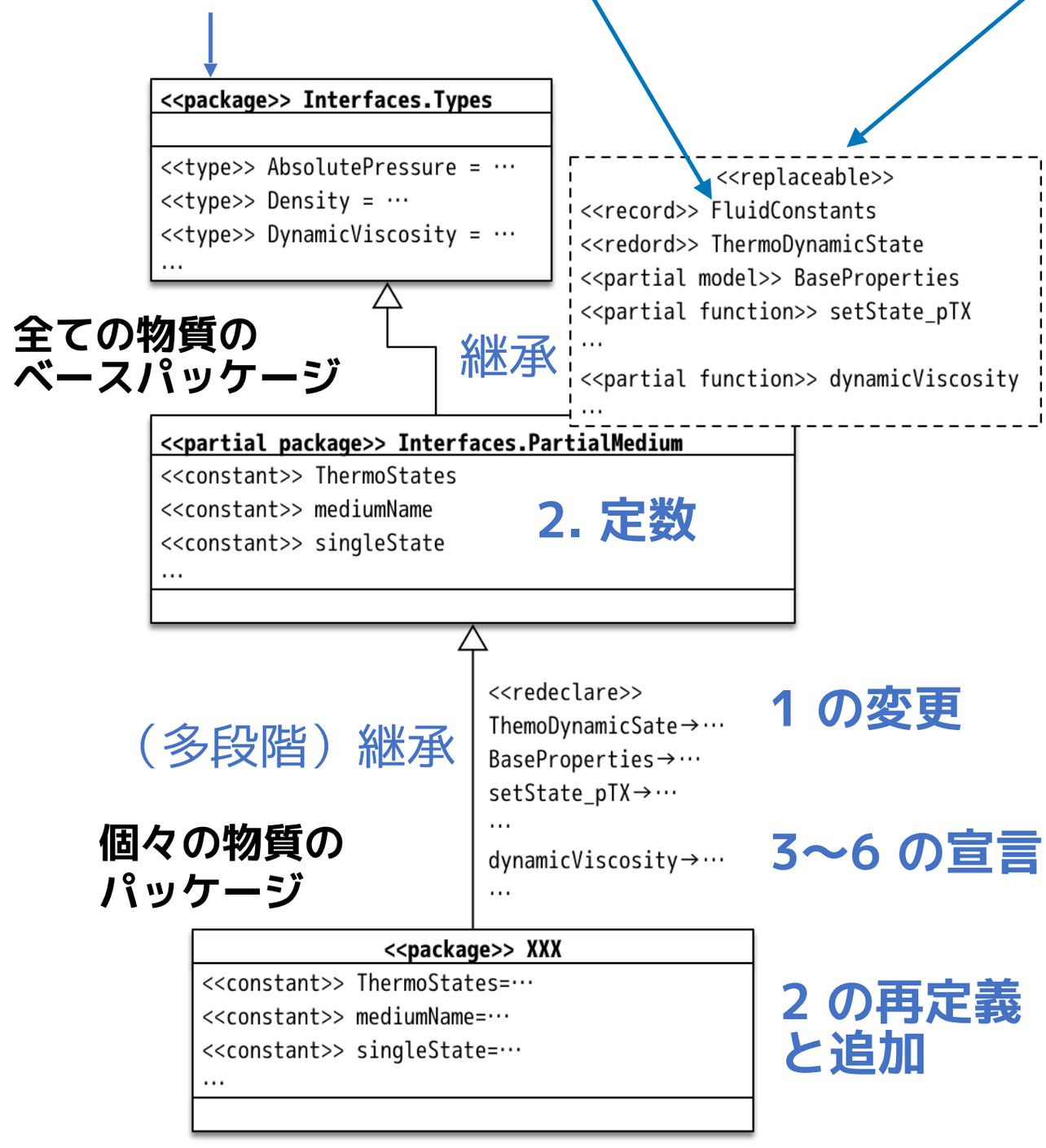
本文書は、Modelica Standard Library に含まれる MoistAir パッケージについて、ソースコードなどから得て調査した内容を紹介するものです。MoistAir は、霧状の領域から 0 ℃ 未満の状態を含む湿り空気全体の熱力学的モデルを提供するものです。モデルの仮定は、完全気体の法則を適用し、水蒸気を除く水の体積を無視します。

Contents

Examples

Modelica.Media の基本構成

1. Types + FluidConstantsレコード



3〜6.
交換可能な record,
model, partial function, …

Medium (媒体物質モデル)
の構成要素

1. **Types (物性値の型)** +
   FluidConstants 定数レコード

2. **Constants (定数)**
   物質名、成分名、参照状態
   デフォルト状態などを示す定数

3. **ThermodynamicState**
   熱力学的状態を表すレコード

4. **BaseProperties**
   基本物性モデル

5. **setState_XXX**
   熱力学的状態
   (ThermodynamicState)
   を返す関数

6. **Property Functions (Optional)**
   粘性率、熱伝導率などの物性値関数

クラス一覧

MoistAir の継承関係

MixtureGasNasa(理想気体混合物)と共通する概念の部分

A **PartialMedium**
物質

B **PartialMixtureMedium**
混合物

C **PartialCondensingGases**
凝縮性のある気体

D **MoistAir**
湿り空気

ParitialMixtureMedium で確定した record

- FluidConstants
- ThermodynamicState

ParitialMixtureMedium で導入された関数

- moleToMassFractions
- massToMoleFractions

モル分率 ⟷ 質量分率

- *gasConstant(state)*

気体定数

PartialCondensingGases で導入される関数

- *saturationPressure(Tsat)*
- *enthalpyOfVaporization(T)*
- *enthalpyOfLiquid(T)*
- *enthalpyOfGas(T,X)*
- *enthalpyOfCondensingGas(T)*
- *enthalpyOfNonCondensingGas(T)*

飽和蒸気圧や構成要素のエンタルピなど

1. Type と FluidConstants　物性値や定数の型

変数や定数の型の宣言の変更部分

PartialMixtureMedium の継承部分

```
partial package PartialMixtureMedium
  "Base class for pure substances of several chemical substances"
  extends PartialMedium(redeclare replaceable record FluidConstants =
      Modelica.Media.Interfaces.Types.IdealGas.FluidConstants);
```

理想気体の FluidConstants レコードを使用する。

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 1.II, p.12

PartialCondensingGasesの継承部分

```
package MoistAir "Air: Moist air model (190 ... 647 K)"
  extends Interfaces.PartialCondensingGases(
    mediumName="Moist air",
    substanceNames={"water","air"},
    final reducedX=true,
    final singleState=false,
    reference_X={0.01,0.99},
    fluidConstants={IdealGases.Common.FluidData.H2O,IdealGases.Common.FluidData.N2},
    Temperature(min=190, max=647));
```

適用可能な温度範囲は 190 [K] から 647 [K]までである。
(-83.15 [degC] から373.85 [degC] まで)

クラス一覧

2. 定数 (Constants)

- 物質名と熱力学的変数特性
- 成分物質の情報
- 微小物質など付加的物質の情報
- reference state
- default state

定数名	型	設定値	
mediumName	String	**Moist air**	物質名
ThermoStates	IndependentVariables	**pTX**	T, pT, ph, phX, pTX, dTXのいずれか
singleState	Boolean	**false**	trueなら物性値が圧力に依存しない
substanceNames[:]	String	**{“water”,”air”}**	成分名の配列
reducedX	Boolean	**true**	trueなら質量分率の和が1
fixedX	Boolean	false	trueならX=reference_X
nS	Integer	size(substanceNames, 1) **= 2**	混合物の成分数
nX	Integer	nS **= 2**	質量分率の要素数
nXi	Integer	if fixedX then 0 else if reducedX nS -1 else nS **= 1**	独立な質量分率の要素数
extraPropertiesNames[:]	String	fill(“”,0)	付加的物質名(微小物質)
nC	Integer	size(extraPropertiesNames, 1)	付加的物質の成分数
C_nominal[nC]	Real	1.0e-6*ones(nC)	付加的物質の濃度
reference_p	AbsolutePressure	101325 [Pa]	圧力の参照値（基準値）
reference_T	Temperature	298.15 [K]	温度の参照値（基準値）
reference_X[nX]	MassFraction	fill(1/nX, nX)	質量分率の参照値
p_defalut	AbsolutePressure	101325 [Pa]	圧力のデフォルト値
T_default	Temperature	Conversions.from_degC(20)	温度のデフォルト値
h_default	SpecificEnthalpy	specifixEnthalpy_pTX(p_default, T_default, X_default)	比エンタルピのデフォルト値
X_default[nX]	MassFraction	reference_X	質量分率のデフォルト値
fluidConstants[nS]	FluidConstants	**{FluidData.H2O, FluidData.N2}**	成分物質のIUPAC命名、化学式など
Water	**Integer**	**1**	水の成分番号
Air	**Integer**	**2**	乾燥空気の成分番号
k_mair	**Real**	**steam.MM/dryair.MM**	水蒸気と乾燥空気のモル質量の比
dryair	**DataRecord**	**SingleGasesData.Air**	乾燥空気の熱物性データ
steam	**DataRecord**	**SingleGasesData.H2O**	水蒸気の熱物性データ
MMX	**MolarMass[2]**	**{steam.MM,dryair.MM}**	

MoistAir で導入された定数

湿り空気の物質的成分 (components of Moist Air)

substanceNames = {“water”, “air”}　成分数 nS=2

水と乾燥空気 (water and dryair)

モル質量 (molar mass)

MMX = {steam.MM, dryair.MM}

モル比と質量比の変換係数 (translation coefficient)

$$k_{mair} \equiv \frac{steam.MM}{dryair.MM}$$

$steam.MM$ [kg/mol]: 水のモル質量

$dryair.MM$ [kg/mol]: 乾燥空気のモル質量

水蒸気と空気の分圧比 = モル比 (molar ratio)

$$\frac{p_{steam}}{p_{air}}, \quad k_{mair} = \frac{X_{steam}}{X_{air}}$$

質量比（mass ratio)

fluidConstants[1] = idealGases.Common.FluidData.H2O

```
  constant Modelica.Media.Interfaces.Types.IdealGas.FluidConstants H2O(
                       chemicalFormula =         "H2O",
                       iupacName =               "oxidane",
                       structureFormula =        "H2O",
                       casRegistryNumber =       "7732-18-5",
                       meltingPoint =            273.15,
                       normalBoilingPoint =      373.124,
                       criticalTemperature =     647.096,
                       criticalPressure =        220.64e5,
                       criticalMolarVolume =      55.95e-6,
                       acentricFactor =            0.344,
                       dipoleMoment =              1.8,
                       molarMass =               SingleGasesData.H2O.MM,
                       hasDipoleMoment =        true,
                       hasIdealGasHeatCapacity=true,
                       hasCriticalData =        true,
                       hasAcentricFactor =      true);
```

fluidConstants[2] = idealGases.Common.FluidData.N2

```
  constant Modelica.Media.Interfaces.Types.IdealGas.FluidConstants N2(
                       chemicalFormula =         "N2",
                       iupacName =               "unknown",
                       structureFormula =        "unknown",
                       casRegistryNumber =       "7727-37-9",
                       meltingPoint =             63.15,
                       normalBoilingPoint =       77.35,
                       criticalTemperature =     126.20,
                       criticalPressure =         33.98e5,
                       criticalMolarVolume =      90.10e-6,
                       acentricFactor =            0.037,
                       dipoleMoment =              0.0,
                       molarMass =               SingleGasesData.N2.MM,
                       hasDipoleMoment =        true,
                       hasIdealGasHeatCapacity=true,
                       hasCriticalData =        true,
                       hasAcentricFactor =      true);
```

FluidConstants のデータ内容の詳細は以下を参照

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 1.II, p.12

dryair = IdealGases.Common.SingleGaseData.Air

```
  constant IdealGases.Common.DataRecord Air(
    name="Air",
    MM=0.0289651159,
    Hf=-4333.833858403446,
    H0=298609.6803431054,
    Tlimit=1000,
    alow={10099.5016,-196.827561,5.00915511,-0.00576101373,1.06685993e-005,-7.94029797e-009,
        2.18523191e-012},
    blow={-176.796731,-3.921504225},
    ahigh={241521.443,-1257.8746,5.14455867,-0.000213854179,7.06522784e-008,-1.07148349e-011,
        6.57780015e-016},
    bhigh={6462.26319,-8.147411905},
    R=287.0512249529787);
```

steam = IdealGases.Common.SingleGaseData.H2O

```
  constant IdealGases.Common.DataRecord H2O(
    name="H2O",
    MM=0.01801528,
    Hf=-13423382.81725291,
    H0=549760.6476280135,
    Tlimit=1000,
    alow={-39479.6083,575.573102,0.931782653,0.00722271286,-7.34255737e-006,
        4.95504349e-009,-1.336933246e-012},
    blow={-33039.7431,17.24205775},
    ahigh={1034972.096,-2412.698562,4.64611078,0.002291998307,-6.836830479999999e-007,
        9.426468930000001e-011,-4.82238053e-015},
    bhigh={-13842.86509,-7.97814851},
    R=461.5233290850878);
```

乾燥空気と水蒸気の比熱、エンタルピ、エントロピの計算でこれらを使用する。

DataRecord 型データの内容の詳細は以下を参照

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf　2.II 熱物性データ

3. ThermodynamicState レコード

質量分率 X の成分数 nX = 2　　**index**　$Water = 1, Air = 2$

$X[Water]$　**水の質量分率 (mass fraction of water)**

$X[Air]$　**乾燥空気の質量分率 (mass fraction of dry air)**

独立な質量分率 Xi の成分数 nXi = 1

reducedX = true, fixedX = false

$$X[Water] = Xi[Water]$$

$$X[Air] = 1 - Xi[Water]$$

MoistAir.ThermodynamicState

```
  redeclare replaceable record extends ThermodynamicState
    "Thermodynamic state variables"
    AbsolutePressure p "Absolute pressure of medium";
    Temperature T "Temperature of medium";
    MassFraction[nX] X(start=reference_X)
      "Mass fractions (= (component mass)/total mass  m_i/m)";
  end ThermodynamicState;
```

独立な熱力学的変数は $p, T, Xi[Water]$ の３個のみとなる。

ThermodynamicState レコードについては、以下を参照。

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 3.

4. BaseProperties

(1) パラメータと変数

<partial model> PartialMedium.BaseProperties
<parameter> preferredMediumStates = false : Boolearn <parameter> standardOrderComponent = true : Boolean p : InputAbsolutePressure Xi: InputMassFraction[nXi] h: InputSpecificEnthalpy d: Density T: Temperature X: MassFraction[nX] u: SpecificInternalEnergy R: SpecificHeatCapacity MM: MolarMass state: Thermodynamic state
<input connector> InputAbsolutePressure <input connector> InputSpecificEnthalpy <input connector> InputMassFraction equation(Xi, X)

熱力学的状態変数、
ガス定数、モル質量
などの関係を表すmodel

<model> MoistAir.BaseProperties
x_water: MassFraction phi: Real <protected> X_liquid: MassFraction <protected> X_steam: MassFraction <protected> X_air: MassFraction <protected> X_sat: MassFraction <protected> x_sat: MassFraction <protected> p_steam_sat: AbsolutePressure
equation(X_sat, X_liquid, X_steam, X_air, p_steam_sat) equation(x_sat, x_water, phi) equation(MM, h, R, u, d, p, T, X)

パラメータ

preferredMediumStatus=true：状態変数選択
standardOrderComponent: = true 質量分率正規化

PartialMedium の BasePropertiesの変数

p	圧力
T	温度
$X = \mathrm{X[nX]}$	質量分率
h	比エンタルピ
$\rho = \mathrm{d}$	密度
u	内部エネルギー
$X_i = \mathrm{Xi[nXi]}$	独立な成分の質量分率
R	気体定数
$M = \mathrm{MM}$	モル質量
$state$	ThermodynamicState, 関接続係

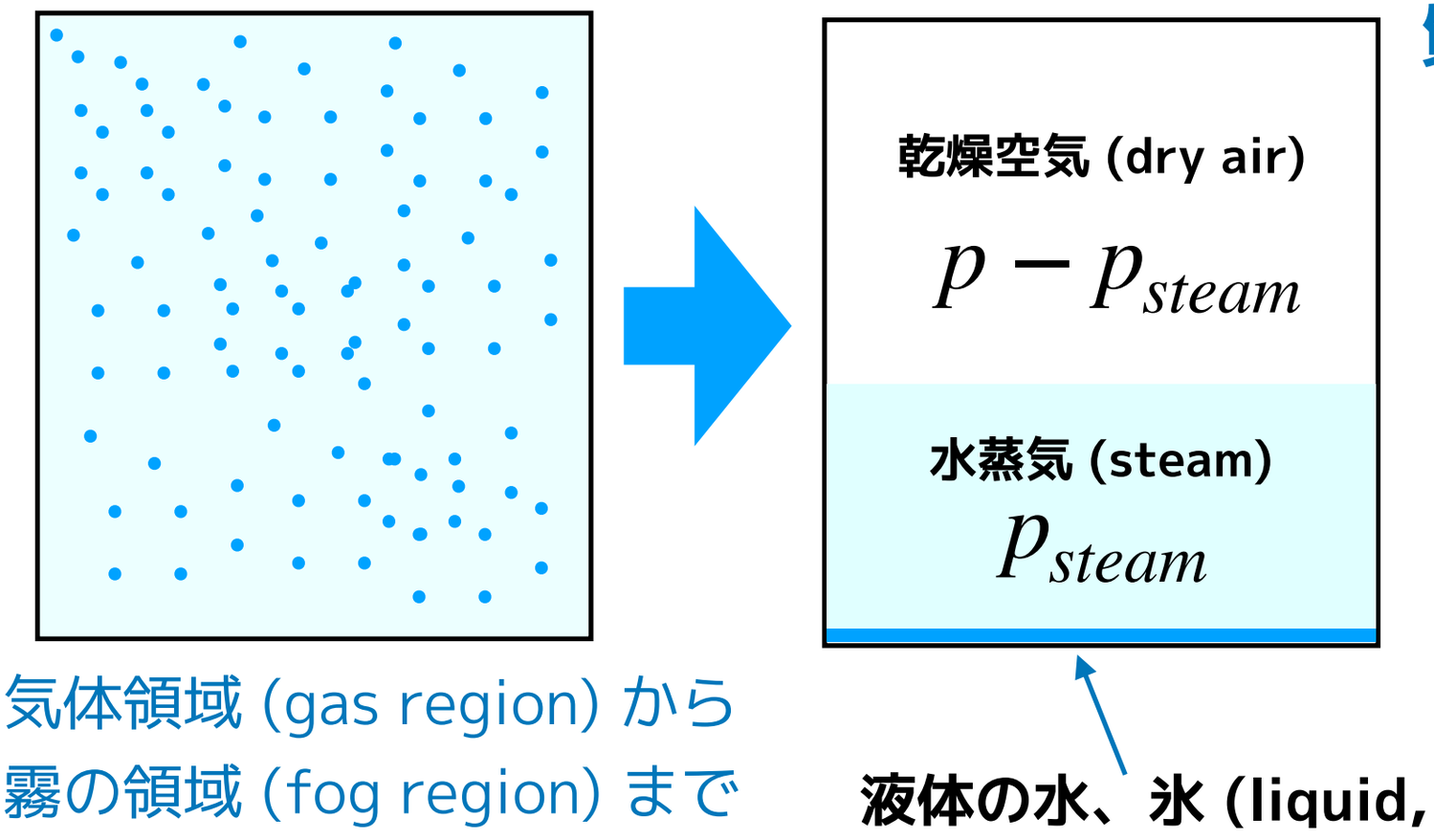
MoistAir の　BaseProperties で追加される変数

$x_{water} = \mathrm{x\_water}$	水質量/乾燥空気質量
$x_{sat} = \mathrm{x\_sat}$	飽和蒸気質量/乾燥空気質量
$\varphi = \mathrm{phi}$	相対湿度 (relative humidity)
$X_{liquid} = \mathrm{X\_liquid}$	水(液＋固)質量/湿り空気質量
$X_{steam} = \mathrm{X\_steam}$	水蒸気質量/湿り空気質量
$X_{air} = \mathrm{X\_air}$	乾燥空気質量/湿り空気質量
$X_{sat} = \mathrm{X\_sat}$	飽和蒸気質量/湿り空気質量
$p_{steam\_sat} = \mathrm{p\_steam\_sat}$	飽和蒸気圧

(2) 湿り空気の構成要素と静的状態変数選択

① 構成要素(components)

湿り空気(moist air)　構成要素(components)



気体領域 (gas region) から
霧の領域 (fog region) まで

質量分率(mass fraction)

$$X_{air} = 1 - Xi[Water]$$

$$\left.\begin{matrix} X_{steam} \\ X_{liquid} \end{matrix}\right] \; Xi[Water]$$

② 独立な熱力学的状態変数(independent state variables)

$p, T, Xi[Water]$ の３個のみ

X_{steam}, X_{liquid}
の割合は p, T に依存する。

③ 静的状態変数選択(Static State Selection)のための条件

A. preferredMediumStates = true のとき **p, T, Xi[Water]** の **stateSelect = prefer**

MoistAir.BasePropertiesの継承部分

```
model extends BaseProperties(
    T(stateSelect=if preferredMediumStates then StateSelect.prefer else StateSelect.default),
    p(stateSelect=if preferredMediumStates then StateSelect.prefer else StateSelect.default),
    Xi(each stateSelect=if preferredMediumStates then StateSelect.prefer else StateSelect.default),
    final standardOrderComponents=true) "Moist air base properties record"
```

B. h, u, d を p, T, Xi[Water] の関数で表す。**BaseProperties の equation より抜粋**

```
  h = specificEnthalpy_pTX(
        p,
        T,
        Xi);
R = dryair.R*(X_air/(1 - X_liquid)) + steam.R*X_steam/(1 - X_liquid);
//
u = h - R*T;
d = p/(R*T);
/* Note, u and d are computed under the assumption that the volume of the liquid
     water is negligible with respect to the volume of air and of steam
  */
state.p = p;
state.T = T;
state.X = X;
```

h は Modelica 関数で表す

state 変数との接続

乾燥空気の質量分率　　水蒸気の質量分率

$$R = R_{air}\frac{X_{dryair}}{1 - X_{liquid}} + R_{steam}\frac{X_{steam}}{1 - X_{liquid}}$$

気体の質量分率

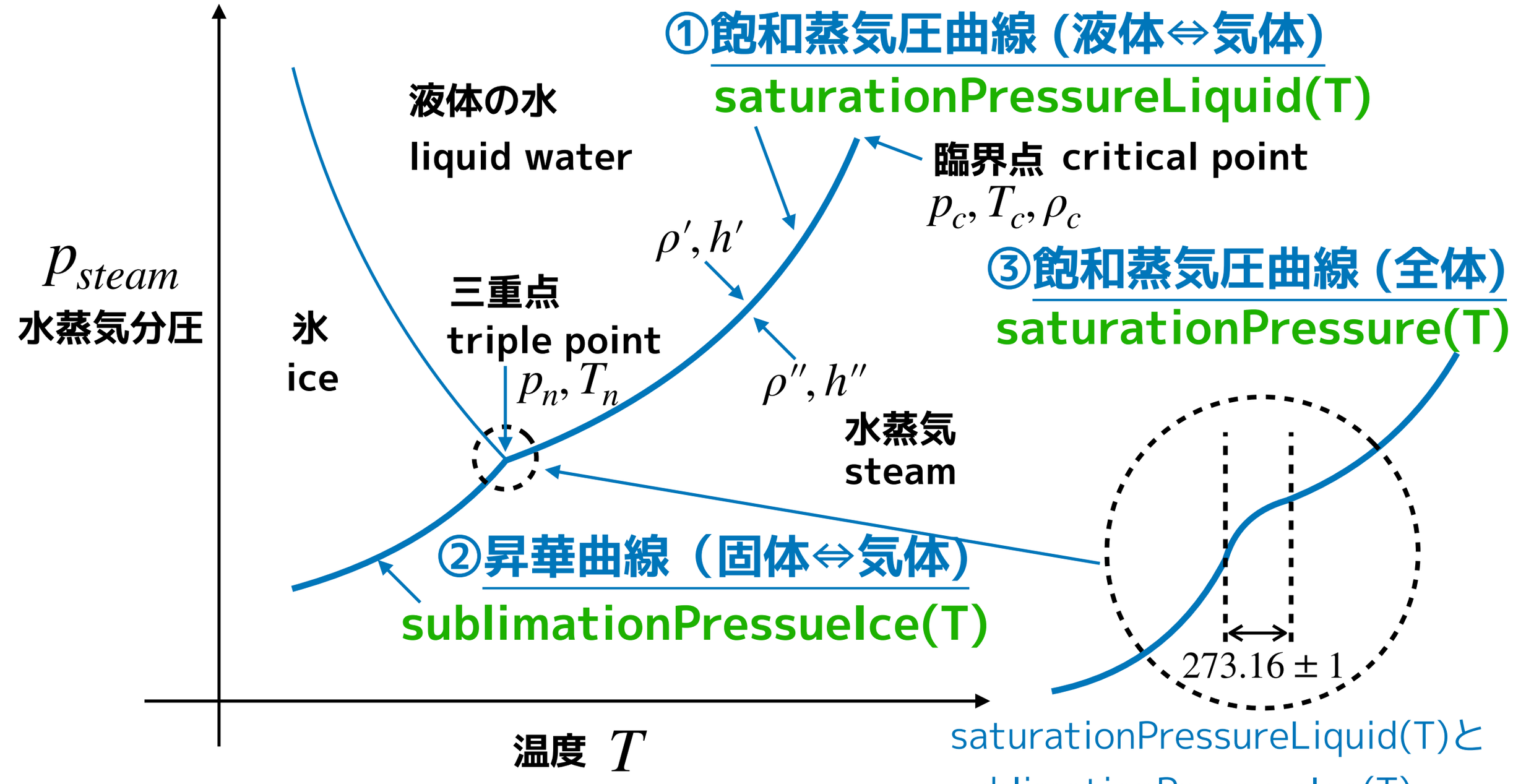
$$pv = RT \quad \text{状態方程式}$$
$$h \equiv u + pv \quad \text{エンタルピ}$$

$$u = h - pv = h - RT$$
$$\rho = \frac{1}{v} = \frac{p}{RT}$$

静的状態変数選択の意味については以下を参照

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 4.I

(3) 飽和蒸気圧曲線 (saturation pressure)



saturationPressureLiquid(T)と
sublimationPressureIce(T)
を滑らかにつなぐ

④BaseProperties の飽和蒸気圧

$$p_{steam\_sat} = \min(\text{saturationPressure}(T), 0.999p)$$

蒸気圧が湿り空気の圧力より小さくなるようにリミッターがかかっている。

飽和状態の水に関する文献[1] (液体⇔気体)

[1] W Wagner, A Pruss: "International equations for the saturation properties of ordinary water substance. Reviesed According to a International Temperature Scale of 1990. Addendum to J. Phys. Chem. Ref. Dat 16, 893(1987)", J. Phys. Chem. Ref. Data 22, 783 (1993)

(a) 飽和蒸気圧 p と臨界圧力 p_c の関係

無次元温度

$$\tau \equiv 1 - \theta = 1 - \frac{T}{T_c}$$

$$\ln\left(\frac{p}{p_c}\right) = \frac{T_c}{T}\left[a_1\tau + a_2\tau^{1.5} + a_3\tau^3 + a_4\tau^{3.5} + a_5\tau^4 + a_6\tau^{7.5}\right], \qquad \textbf{(1)}$$

$$a_1 = -7.85951783, \quad a_2 = 1.84408259, \quad a_3 = -11.7866497,$$
$$a_4 = 22.6807411, \quad a_5 = -15.9618719, \quad a_6 = 1.80122502$$

(b) 飽和水密度 ρ' と臨界密度 ρ_c の関係

$$\frac{\rho'}{\rho_c} = 1 + b_1\tau^{1/3} + b_2\tau^{2/3} + b_3\tau^{5/3} + b_4\tau^{16/3} + b_5\tau^{43/3} + b_6\tau^{110/3} \qquad \textbf{(2)}$$

$$b_1 = 1.99274064, \quad b_2 = 1.09965342, \quad b_3 = -0.510839303,$$
$$b_4 = -1.75493479, \quad b_5 = -45.5170352, \quad b_6 = -6.74694450 \times 10^5$$

$$m_1 = 1/3,\ m_2 = 2/3,\ m_3 = 5/3,\ m_4 = 16/3,\ m_5 = 43/3,\ m_6 = 110/3$$

(c) 飽和蒸気密度 ρ'' と臨界密度 ρ_c の関係 (Saturation Pressure)

$$\ln\left(\frac{\rho''}{\rho_c}\right) = c_1\tau^{2/6} + c_2\tau^{4/6} + c_3\tau^{8/6} + c_4\tau^{18/6} + c_5\tau^{37/6} + c_6\tau^{71/6} \tag{3}$$

$$c_1 = -2.03150240, \quad c_2 = -2.68302940 \quad c_3 = -5.38626492,$$
$$c_4 = --17.2991605, \quad c_5 = -44.7586581, \quad c_6 = -63.9201063$$

(d) 飽和水の比エンタルピ

(Specific enthalpy of the saturated liquid)

$$h' = \alpha + \frac{T}{\rho'}\frac{dp}{dT} \tag{6}$$

(e) 飽和蒸気の比エンタルピ

(Specific enthalpy of the saturated vapor)

$$h'' = \alpha + \frac{T}{\rho''}\frac{dp}{dT} \tag{7}$$

飽和状態の水に関する文献[2][3] (固体⇔気体)

[2] W Wagner, A Saul, A Pruss: "International equations for the pressure along the melting and along the sublimation curve of ordinary water substance", equation 3.5

[3] W.Wagner, A.Pruß, _ The IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use, J.Phys.Chem.Ref.Data, Vol.31,No. 2,2002. eq.2.21 (http://www.teos-10.org/pubs/Wagner_and_Pruss_2002.pdf)

(f) 昇華曲線の圧力 (sublimation pressure)

$$\ln \frac{P_{subl}}{P_n} = -13.928169\left(1-\theta^{-1.5}\right) + 34.7078238\left(1-\theta^{-1.25}\right)$$

$$\theta = \frac{T}{T_n},\ T_n = 273.16$$

①　飽和蒸気圧曲線(液体⇔気体)

(a) $\Leftrightarrow p_{sat} = \exp\left((a_1\tau^{n_1} + a_2\tau^{n_2} + a_3\tau^{n_3} + a_4\tau^{n_4} + a_5\tau^{n_5} + a_6\tau^{n_6})\frac{T_c}{T_{sat}} \right) p_c$

$$\tau \equiv 1 - \theta = 1 - \frac{T_{sat}}{T_c}$$

$$n_1 = 1,\ n_2 = 1.5,\ n_3 = 3,\ n_4 = 3.5,\ n_5 = 4,\ n_5 = 7.5$$

MoistAir.satulationPressureLiquid(Tsat)

protected 変数と algorithm の抜粋

r1 = τ

```
protected
  SI.Temperature Tcritical=647.096 "Critical temperature";
  SI.AbsolutePressure pcritical=22.064e6 "Critical pressure";
  Real r1=(1 - Tsat/Tcritical) "Common subexpression";
  Real a[:]={-7.85951783,1.84408259,-11.7866497,22.6807411,-15.9618719,
      1.80122502} "Coefficients a[:]";
  Real n[:]={1.0,1.5,3.0,3.5,4.0,7.5} "Coefficients n[:]";
algorithm
  psat := exp(((a[1]*r1^n[1] + a[2]*r1^n[2] + a[3]*r1^n[3] + a[4]*r1^n[4]
     + a[5]*r1^n[5] + a[6]*r1^n[6])*Tcritical)/Tsat)*pcritical;
  annotation ( …);
end saturationPressureLiquid;
```

①’ 飽和蒸気圧曲線(液体⇔気体)の温度微分

$$p_{sat} = \exp\left(r_2 \frac{T_c}{T_{sat}}\right) p_c, \quad r_2(\tau) \equiv a_1\tau^{n_1} + a_2\tau^{n_2} + a_3\tau^{n_3} + a_4\tau^{n_4} + a_5\tau^{n_5} + a_6\tau^{n_6}$$

$$\Rightarrow dp_{sat} = \exp\left(r2 \frac{T_c}{T_{sat}}\right) p_c \left\{ \left(\frac{dr_2}{d\tau} d\tau\right) \frac{T_c}{T_{sat}} - r_2 \frac{T_c}{T_{sat}^2} dT_{sat} \right\}$$

$$\frac{dr_2}{d\tau} d\tau = a_1 n_1 \tau^{(n_1-1)} d\tau + a_2 n_2 \tau^{(n_2-1)} d\tau + a_3 n_3 \tau^{(n_3-1)} d\tau + a_4 n_4 \tau^{(n_4-1)} d\tau + a_5 n_5 \tau^{(n_5-1)} d\tau + a_6 n_6 \tau^{(n_6-1)} d\tau$$

MoistAir.satulationPressureLiquid_der(Tsat, dTsat)

protected 変数と algorithm の抜粋

```
protected
  SI.Temperature Tcritical=647.096 "Critical temperature";
  SI.AbsolutePressure pcritical=22.064e6 "Critical pressure";
  Real r1=(1 - Tsat/Tcritical) "Common subexpression 1";
  Real r1_der=-1/Tcritical*dTsat "Derivative of common subexpression 1";
  Real a[:]={-7.85951783,1.84408259,-11.7866497,22.6807411,-15.9618719,
      1.80122502} "Coefficients a[:]";
  Real n[:]={1.0,1.5,3.0,3.5,4.0,7.5} "Coefficients n[:]";
  Real r2=(a[1]*r1^n[1] + a[2]*r1^n[2] + a[3]*r1^n[3] + a[4]*r1^n[4] + a[5]
      *r1^n[5] + a[6]*r1^n[6]) "Common subexpression 2";
algorithm
...
 psat_der := exp((r2*Tcritical)/Tsat)*pcritical*((a[1]*(r1^(n[1] - 1)*n[1]
    *r1_der) + a[2]*(r1^(n[2] - 1)*n[2]*r1_der) + a[3]*(r1^(n[3] - 1)*n[3]*
    r1_der) + a[4]*(r1^(n[4] - 1)*n[4]*r1_der) + a[5]*(r1^(n[5] - 1)*n[5]*
    r1_der) + a[6]*(r1^(n[6] - 1)*n[6]*r1_der))*Tcritical/Tsat - r2*
    Tcritical*dTsat/Tsat^2);
```

r1= $\tau = 1 - \frac{T_{sat}}{T_c}$

r1_der $= d\tau = -\frac{1}{T_c} dT_{sat}$

②　昇華曲線(固体⇔気体)

(f)　$\Leftrightarrow p_{subl} = \exp\left(-13.9281690(1-\theta^{-1.5}) + 34.7078238(1-\theta^{-1.25})\right) P_n$

$\Leftrightarrow p_{subl} = \exp\left(a_1 - \ a1\theta^{n_1} + a_2 - a_2\theta^{n_2}\right) P_n$

$$\theta = \frac{T_{sat}}{T_n},\ T_n = 273.16$$

$$a_1 = -13.9281690,\ a_2 = 34.7078238,\ n1 = -1.5,\ n_2 = -1.25$$

MoistAir.sublimationPressureIce(Tsat)

protected 変数と algorithm の抜粋

```
protected
  SI.Temperature Ttriple=273.16 "Triple point temperature";
  SI.AbsolutePressure ptriple=611.657 "Triple point pressure";
  Real r1=Tsat/Ttriple "Common subexpression";
  Real a[:]={-13.9281690,34.7078238} "Coefficients a[:]";
  Real n[:]={-1.5,-1.25} "Coefficients n[:]";
algorithm
  psat := exp(a[1] - a[1]*r1^n[1] + a[2] - a[2]*r1^n[2])*ptriple;
  annotation ( …);
end sublimationPressureIce;
```

②’ 昇華曲線(固体⇔気体)の温度微分

$$p_{subl} = \exp\left(a_1 - \ a1\theta^{n_1} + a_2 - a_2\theta^{n_2}\right) P_n$$

$$dp_{subl} = \exp\left(a_1 - \ a1\theta^{n_1} + a_2 - a_2\theta^{n_2}\right) P_n(-a_1 n_1 \theta^{n_1 - 1} d\theta - a_2 n_2 \theta^{n2-1} d\theta)$$

$$\theta = \frac{T_{sat}}{T_n},\ T_n = 273.16, \quad d\theta = \frac{1}{T_{triple}} dT_{sat}$$

MoistAir.sublimationPressureIce_der(Tsat, dTsat)

protected 変数と algorithm の抜粋

```
protected
  SI.Temperature Ttriple=273.16 "Triple point temperature";
  SI.AbsolutePressure ptriple=611.657 "Triple point pressure";
  Real r1=Tsat/Ttriple "Common subexpression 1";
  Real r1_der=dTsat/Ttriple "Derivative of common subexpression 1";
  Real a[:]={-13.9281690,34.7078238} "Coefficients a[:]";
  Real n[:]={-1.5,-1.25} "Coefficients n[:]";
algorithm
  //psat := exp(a[1] - a[1]*r1^n[1] + a[2] - a[2]*r1^n[2]) * ptriple;
  psat_der := exp(a[1] - a[1]*r1^n[1] + a[2] - a[2]*r1^n[2])*ptriple*(-(a[1]
    *(r1^(n[1] - 1)*n[1]*r1_der)) - (a[2]*(r1^(n[2] - 1)*n[2]*r1_der)));
```

③ 飽和蒸気圧曲線 (全体)

①,②を273.16±1 K で滑らかにつなぐ

MoistAir.saturationPressure(Tsat)

$$\text{saturationPressure}(T_{sat}) = \text{spliceFunction}(\text{pos}, \text{neg}, x, \Delta x)$$
$$= y\ \text{pos} + (1 - y)\text{neg}$$
$$x = T_{sat} - 273.16 \quad \Delta x = 1.0$$
$$\text{pos} = \text{saturationPressureLiquid}(T_{sat}) \quad ①$$
$$\text{neg} = \text{sublimationPressureIce}(T_{sat}) \quad ②$$

algorithm の抜粋

```
algorithm
  psat := Utilities.spliceFunction(
        saturationPressureLiquid(Tsat),
        sublimationPressureIce(Tsat),
        Tsat - 273.16,
        1.0);
```

MoistAir.Utilities.spliceFunction　つなぐための補完関数

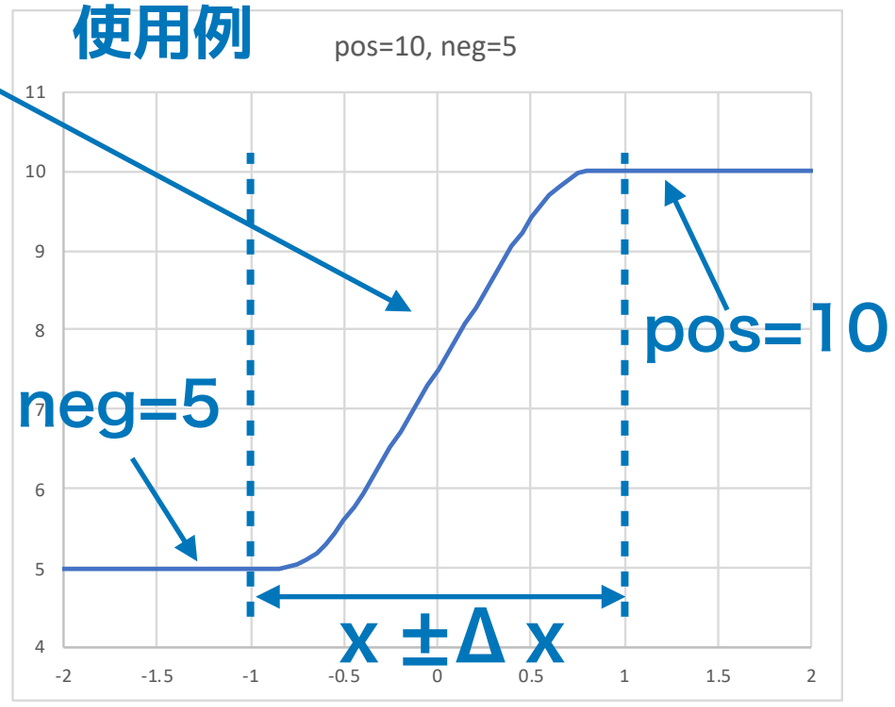
$$\text{spliceFunction}(\text{pos}, \text{neg}, x, \Delta x = 1) \equiv y\ \text{pos} + (1 - y)\ \text{neg} \quad (a)$$

$$\text{scaledX1} = \frac{x}{\Delta x},$$

$$\text{scaledX} = \arcsin 1 \cdot \text{scaledX1} = \frac{\pi}{2}\frac{x}{\Delta x}$$

$$y = \begin{cases} 0, & \text{scaledX1} < -0.999999999 \\ 1, & \text{scaledX1} > 0.999999999 \\ \frac{1}{2}\left[\tanh\left(\tan\left(\text{scaledX}\right)\right) + 1\right], & else \end{cases} \quad (b)$$

使用例



③’ 飽和蒸気圧曲線（全体）の微分

MoistAir.saturationPressure_der(Tsat, dTsat)

$$\text{saturationPressure\_der}(T_{sat}, dT_{sat})$$

algorithm の抜粋

```
psat_der := Utilities.spliceFunction_der(
      saturationPressureLiquid(Tsat),
      sublimationPressureIce(Tsat),
      Tsat - 273.16,
      1.0,
      saturationPressureLiquid_der(Tsat=Tsat, dTsat=dTsat),
      sublimationPressureIce_der(Tsat=Tsat, dTsat=dTsat),
      dTsat,
      0);
```

① 飽和蒸気圧(液体⇔気体)

② 昇華曲線(固体⇔気体)

①’ 飽和蒸気圧(液体⇔気体)の微分

②’ 飽和蒸気圧(液体⇔気体)の微分

spliceFlunciotn(pos,neg,x,dx)の微分

spliceFlunciotn_der(pos,neg,x,dx,dpos,dneg,dx,ddeltax)

を使用する。

spliceFunction の微分

公式　$(\tan x)' = \dfrac{1}{cos^2 x}$　　$(\tanh x)' = \dfrac{1}{\cosh^2 x}$

MoistAir.Utilities.spliceFunction_der (pos, neg,x,deltax=1,dpos,dneg,dx,ddeltax)

splieceFunction

(a)

$$\text{spliceFunction} = y\ \text{pos} + (1 - y)\ \text{neg}$$

(b)

$$y = \begin{cases} 0, & \text{scaledX1} < -0.999999999 \\ 1, & \text{scaledX1} > 0.999999999 \\ \frac{1}{2}\left[\tanh\left(\tan\left(\text{scaledX}\right)\right) + 1\right], & else \end{cases}$$

$$\text{scaledX1} = \frac{x}{\Delta x}$$

$$\text{scaledX} = \text{scaledX1} \cdot \arcsin(1) = \frac{x}{\Delta x}\frac{\pi}{2}$$

$$d\text{scaledX1} = \frac{(dx - \text{scaledX1}\, d\Delta x)}{\Delta x}$$

(a)の微分

$$d(\text{spliceFunction}) = y\ d\text{pos} + (1 - y)\ d\text{neg} + (\text{pos} - \text{neg})dy$$

(b)の微分

$$dy = \begin{cases} 0, & |\text{scaledX1}| \geq 1 \\ d\left(\frac{1}{2}\left[\tanh\left(\tan\left(\text{scaledX}\right)\right) + 1\right]\right) = \dfrac{d\text{scaledX1}(\pi/2)}{2(\cosh(\tan(\text{scaledX}))\cos(\text{scaledX}))^2}, & |\text{scaledX1}| < 1 \end{cases}$$

整理すると…

$$\text{spliceFunction\_der}(pos, neg, \Delta X = 1, dpos, dneg, dx, d\Delta x)$$

$$out = y\ d\text{pos} + (1 - y)\ d\text{neg}$$

$$\text{if } (|\text{scaledX1}| < 1) \text{ then}$$

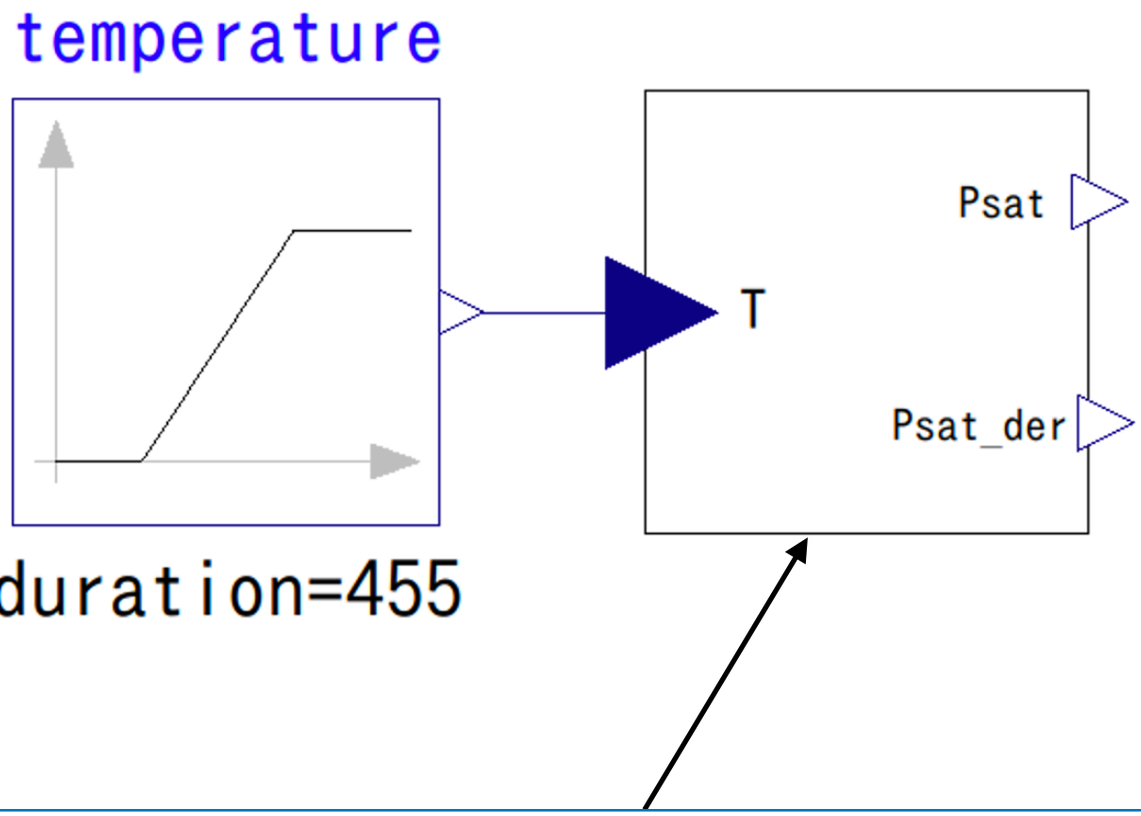
$$out = out + (\text{pos} - \text{neg})\frac{d\text{scaledX1}(\pi/2)}{2(\cosh(\tan(\text{scaledX}))\cos(\text{scaledX}))^2}$$

spliceFunction_der の algorithm

```
algorithm
  scaledX1 := x/deltax;
  scaledX := scaledX1*Modelica.Math.asin(1);
  dscaledX1 := (dx - scaledX1*ddeltax)/deltax;
  if scaledX1 <= -0.99999999999 then
    y := 0;
  elseif scaledX1 >= 0.9999999999 then
    y := 1;
  else
    y := (Modelica.Math.tanh(Modelica.Math.tan(scaledX)) + 1)/2;
  end if;
  out := dpos*y + (1 - y)*dneg;
  if (abs(scaledX1) < 1) then
    out := out + (pos - neg)*dscaledX1*Modelica.Math.asin(1)/2/(
      Modelica.Math.cosh(Modelica.Math.tan(scaledX))*Modelica.Math.cos(
      scaledX))^2;
  end if;
end spliceFunction_der;
```

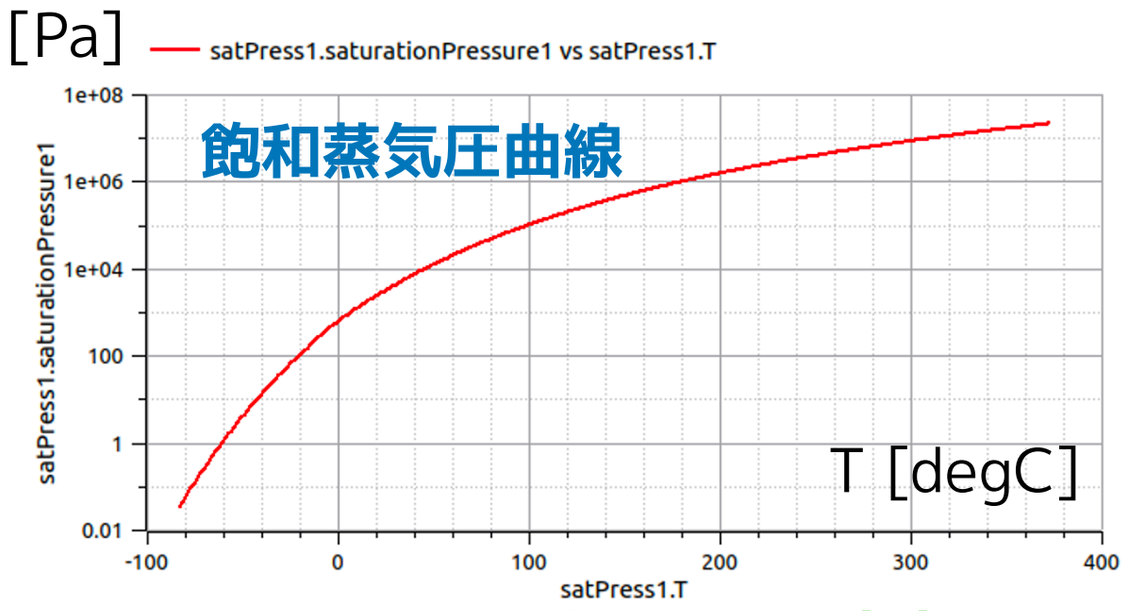
EX.1 飽和蒸気圧曲線を描く

Plot saturation pressure!

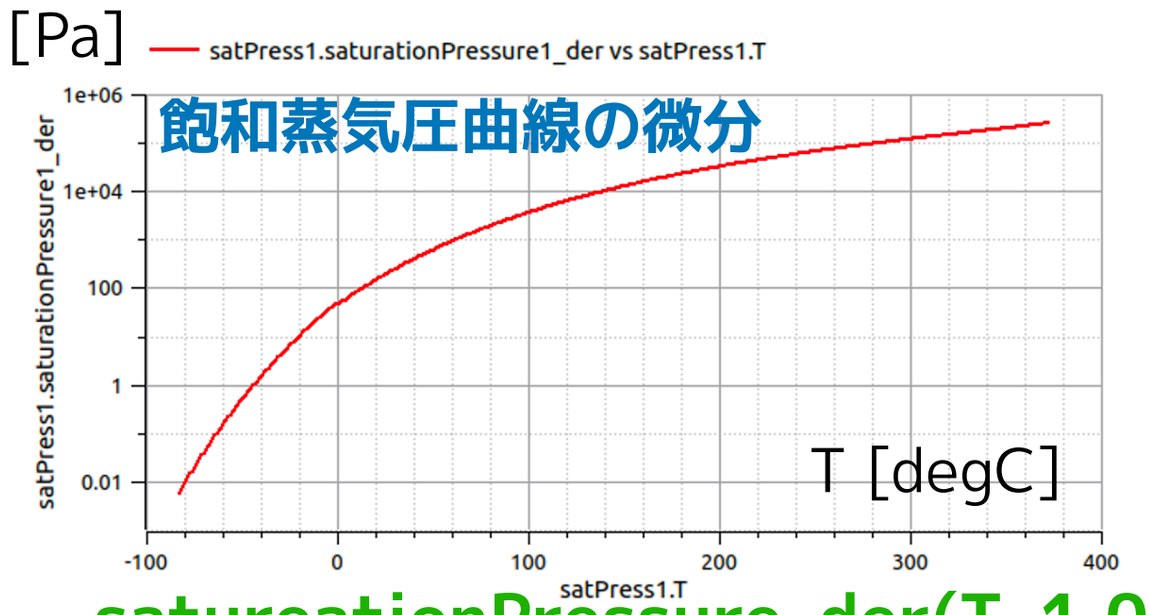



```
model SatPress
  replaceable package Medium = Modelica.Media.Air.MoistAir;
  Modelica.Blocks.Interfaces.RealInput T annotation( …);
  Modelica.Blocks.Interfaces.RealOutput saturationPressure1 annotation( …);
  Modelica.Blocks.Interfaces.RealOutput saturationPressure1_der annotation( …);
equation
  saturationPressure1 = Medium.saturationPressure(T + 273.15);
  saturationPressure1_der = Medium.saturationPressure_der(T + 273.15, 1.0);
  annotation( …);
end SatPress;
```

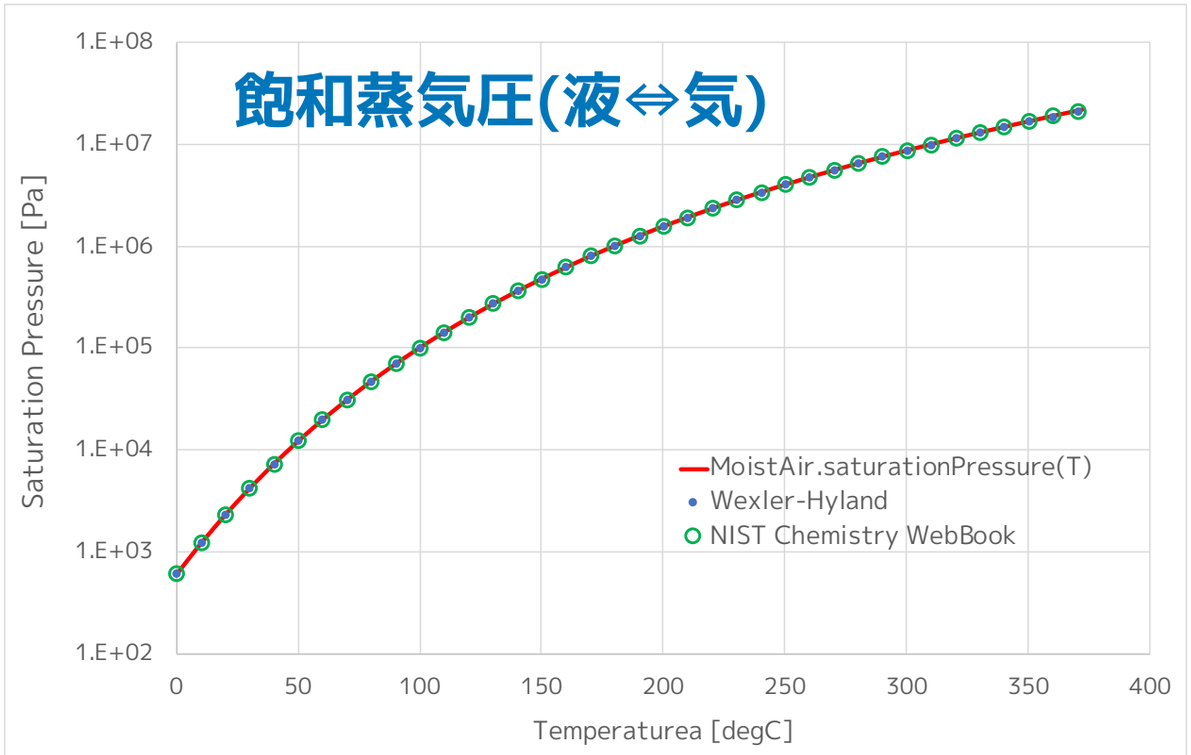
シミュレーション結果

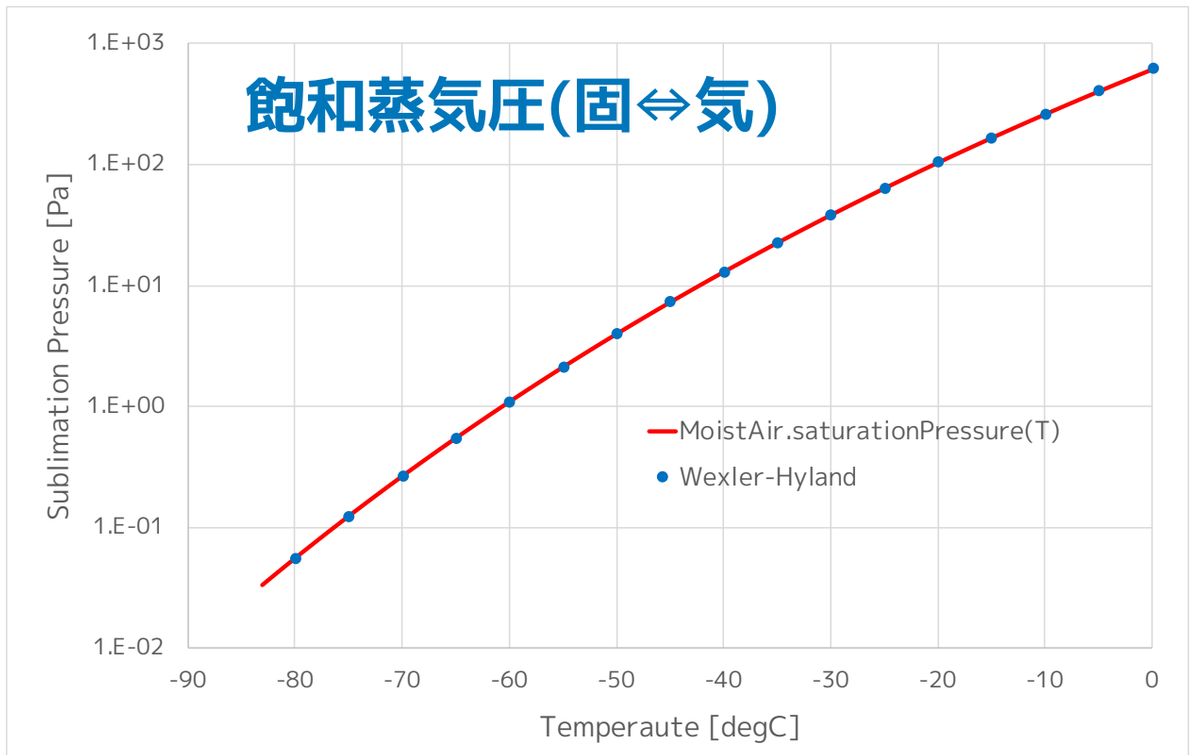


satureationPressure(T)



satureationPressure_der(T, 1.0)





Wexler-Hyland の式は右をを参照した。　http://www.bohra.jp/psychrometric.html

(4) 気化熱 (enthalpy of vaporization)

気化熱 = 飽和蒸気の比エンタルピ - 飽和水の比エンタルピ

文献[1][2]を使用。

(d),(e)　$$\Rightarrow h'' - h' = T\left(\frac{1}{\rho''} - \frac{1}{\rho'}\right)\frac{dp}{dT} = -\frac{(\rho' - \rho'')\exp(r_1)P_c(r_2 + \tau r_1)}{\rho'\rho''\tau}$$

MoistAir.enthalpyOfVaporization(T) より抜粋

```
algorithm
  r0 := -(((dp - dpp)*exp(r1)*pcritical*(r2 + r1*tau))/(dp*dpp*tau))
    "Difference of equations (7) and (6)";
```

(a) $$\Leftrightarrow p = \exp(r_1)p_c,$$

$$\Rightarrow \frac{dp}{dT} = \exp\left(r_1\right)p_c\frac{dr_1}{dT} = exp(r_1)P_c\left(-\frac{r_2 + \tau r_1}{\tau T}\right),\ \tau \equiv 1 - \theta = 1 - \frac{T}{T_c}$$

$$r_2 = a_1 n_1 \tau^{n_1} + a_2 n_2 \tau^{n_2} + a_3 n_3 \tau^{n_3} + a_4 n_4 \tau^{n_4} + a_5 n_5 \tau^{n_5} + a_6 n_6 \tau^{n_6}$$

$$r_1 = \frac{T_c}{T}[a_1\tau^{n_1} + a_2\tau^{n_2} + a_3\tau^{n_3} + a_4\tau^{n_4} + a_5\tau^{n_5} + a_6\tau^{n_6}]$$

MoistAir.enthalpyOfVaporization(T) より抜粋

```
    Real tau=1 - T/Tcritical "Temperature expression";
    Real r1=(a[1]*Tcritical*tau^n[1])/T + (a[2]*Tcritical*tau^n[2])/T + (a[3]
        *Tcritical*tau^n[3])/T + (a[4]*Tcritical*tau^n[4])/T + (a[5]*
        Tcritical*tau^n[5])/T + (a[6]*Tcritical*tau^n[6])/T "Expression 1";
    Real r2=a[1]*n[1]*tau^n[1] + a[2]*n[2]*tau^n[2] + a[3]*n[3]*tau^n[3] + a[
        4]*n[4]*tau^n[4] + a[5]*n[5]*tau^n[5] + a[6]*n[6]*tau^n[6]
      "Expression 2";
```

(b) $\Leftrightarrow \rho' = \rho_c \left(1 + b_1\tau^{m_1} + b_2\tau^{m_2} + b_3\tau^{m_3} + b_4\tau^{m_4} + b_5\tau^{m_5} + b_6\tau^{m_6}\right)$

$$m_1 = 1/3,\ m_2 = 2/3,\ m_3 = 5/3,\ m_4 = 16/3,\ m_5 = 43/3,\ m_6 = 110/3$$

$$b_1 = 1.99274064, \quad b_2 = 1.09965342, \quad b_3 = -0.510839303,$$
$$b_4 = -1.75493479, \quad b_5 = -45.5170352, \quad b_6 = -6.74694450 \times 10^5$$

MoistAir.enthalpyOfVaporization(T) より抜粋

```
    Real dp=dcritical*(1 + b[1]*tau^m[1] + b[2]*tau^m[2] + b[3]*tau^m[3] + b[
        4]*tau^m[4] + b[5]*tau^m[5] + b[6]*tau^m[6])
      "Density of saturated liquid";
```

```
    Real m[:]={1/3,2/3,5/3,16/3,43/3,110/3} "Powers in equation (2)";
    Real b[:]={1.99274064,1.09965342,-0.510839303,-1.75493479,-45.5170352,-6.74694450e5}
      "Coefficients in equation (2) of [1]";
```

(c)

$$\Leftrightarrow \rho'' = \rho_c \exp\left(c_1\tau^{o_1} + c_2\tau^{o_2} + c_3\tau^{o_3} + c_4\tau^{o_4} + c_5\tau^{o_5} + c_6\tau^{o_6}\right)$$

$$c_1 = -2.03150240, \quad c_2 = -2.68302940 \quad c_3 = -5.38626492,$$

$$c_4 = --17.2991605, \quad c_5 = -44.7586581, \quad c_6 = -63.9201063$$

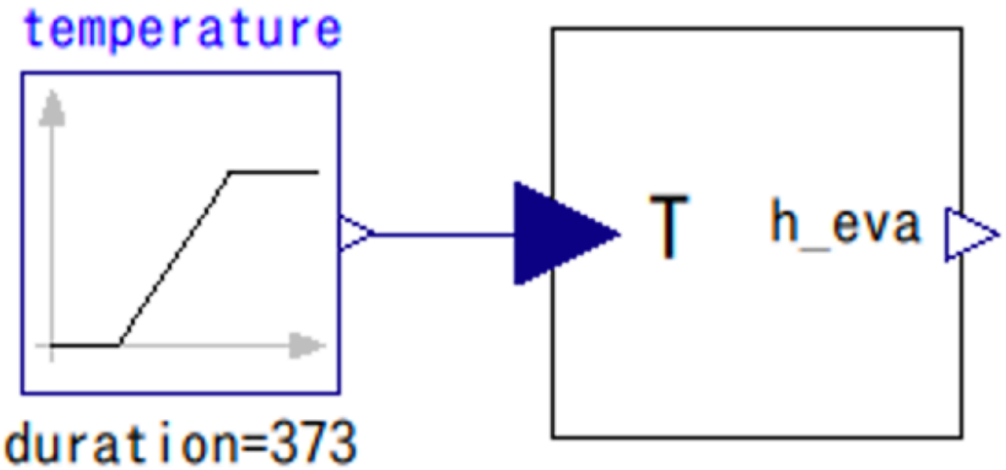
$$o_1 = 2/6,\ o_2 = 4/6,\ o_3 = 8/6,\ o_4 = 18/6,\ o_5 = 37/6,\ o_6 = 71/6$$

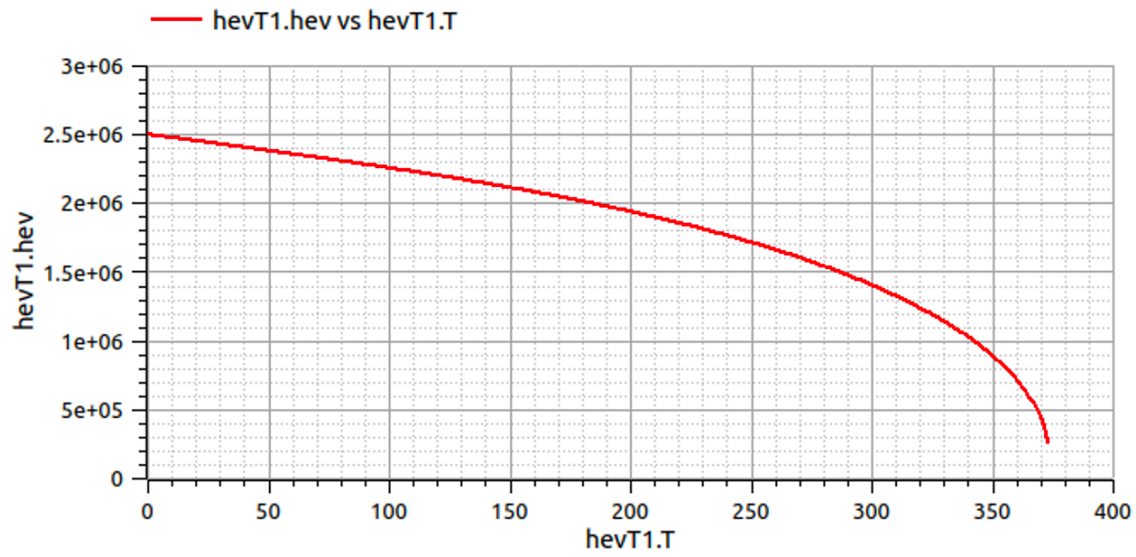
MoistAir.enthalpyOfVaporization(T) より抜粋

```
  Real dpp=dcritical*exp(c[1]*tau^o[1] + c[2]*tau^o[2] + c[3]*tau^o[3] + c[
      4]*tau^o[4] + c[5]*tau^o[5] + c[6]*tau^o[6])
    "Density of saturated vapor";
```

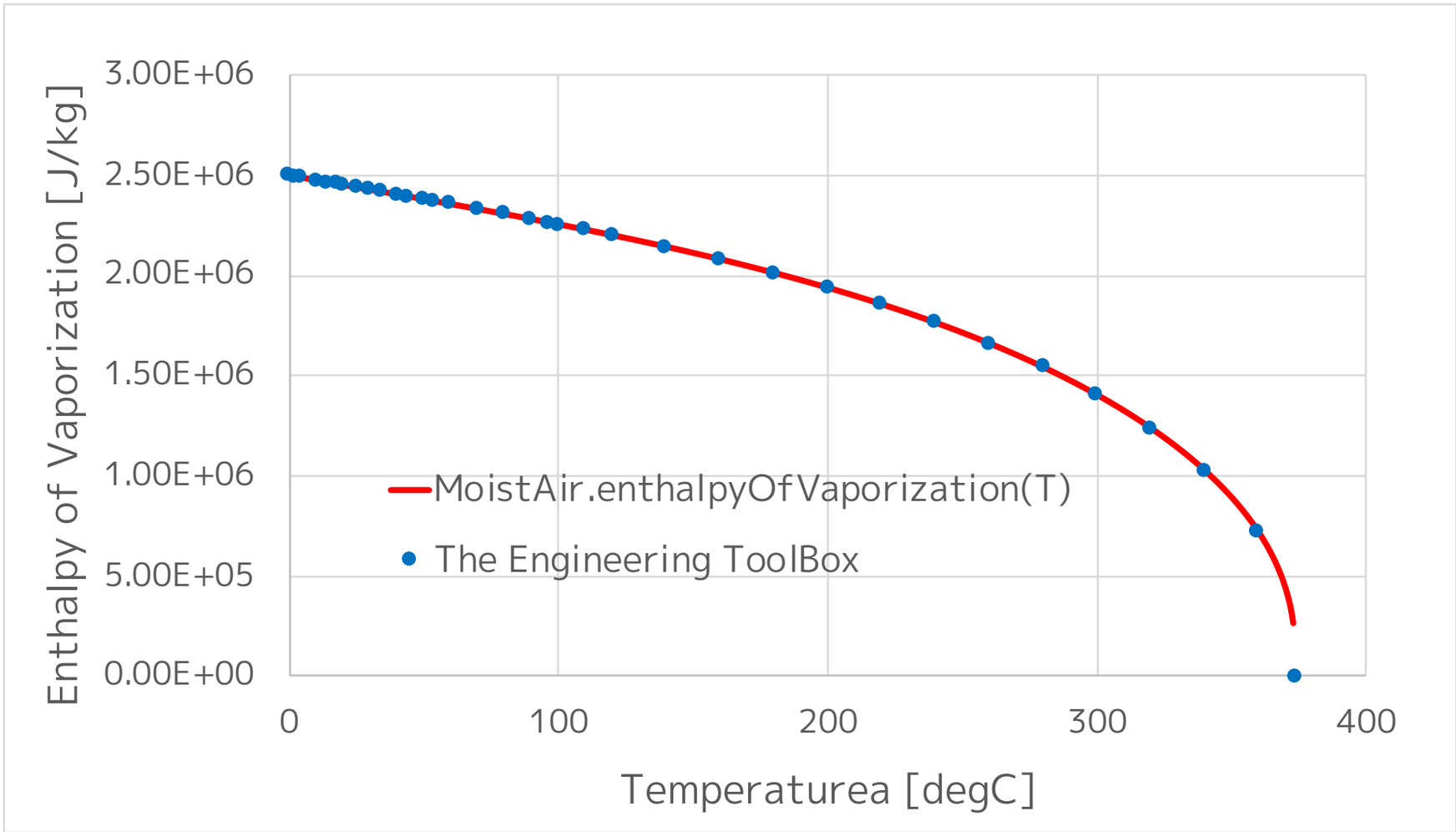
```
  Real o[:]={2/6,4/6,8/6,18/6,37/6,71/6} "Powers in equation (3)";
  Real c[:]={-2.03150240,-2.68302940,-5.38626492,-17.2991605,-44.7586581,-63.9201063}
    "Coefficients in equation (3) of [1]";
```

EX2. 気化熱曲線を描く






```
model HevT
  replaceable package Medium = Modelica.Media.Air.MoistAir;
  Modelica.Blocks.Interfaces.RealInput T annotation( …);
  Modelica.Blocks.Interfaces.RealOutput hev annotation( …);
equation
  hev = Medium.enthalpyOfVaporization(T+273.15);
        annotation( …);
end HevT;
```



The Engineering ToolBox

https://www.engineeringtoolbox.com/water-properties-d_1573.html

(5) 飽和水蒸気の質量分率

湿り空気 1kg に含むことができる水蒸気の質量

① BaseProperties の式

飽和蒸気と乾燥空気の質量比

乾燥空気の質量分率

$$X_{sat} = \frac{p_{steam\_sat}}{p - p_{stream\_sat}} k_{mair} (1 - Xi[Water])$$

- 分母がゼロにならない
- 質量分率が１を超えない

$$X_{sat} = \min\left(\frac{p_{steam\_sat}}{\max(100\varepsilon, p - p_{steam\_sat})} k_{m\_air}(1 - Xi[Water]),\ 1.0 \right)$$

② MoistAir.Xsaturation(state)の計算式

$$X_{sat} = \frac{k_{mair}}{\frac{state.p}{\min(\text{saturationPressure}(state.T), 0.999 state.p)} - 1 + k_{mair}}$$

$$= \frac{p_{steam\_sat}\ steam.MM}{(state.p - p_{steam\_sat})\ dryair.MM + p_{steam\_sat}\ steam.MM}$$

飽和蒸気質量

空気質量

飽和蒸気質量

② massFraction_pTphi(p, T, phi)

圧力 p 温度 T 相対湿度 φ の関数として水蒸気の質量分率を求める。

$$X_{steam} = \frac{k\varphi}{k\varphi + \frac{p}{p_{sat}} - \varphi}$$

相対湿度

$$\varphi = \frac{\frac{p_{steam}}{p}}{\frac{p_{sat}}{p}} = \frac{p_{steam}}{p_{sat}}$$

モル質量比

$$k = \frac{steam.MM}{dryair.MM}$$

$$= \frac{\frac{steam.MM}{dryair.MM}\frac{p_{steam}}{p_{sat}}}{\frac{steam.MM}{dryair.MM}\frac{p_{steam}}{p_{sat}} + \frac{p}{p_{sat}} - \frac{p_{steam}}{p_{sat}}}$$

水蒸気質量比　乾燥空気質量比

$$= \frac{steam.MM\ p_{steam}}{steam.MM\ p_{steam} + dryair.MM(p - p_{steam})}$$

protected 変数と algorithm の抜粋

```
protected
  constant Real k=0.621964713077499 "Ratio of molar masses";
  AbsolutePressure psat=saturationPressure(T) "Saturation pressure";
algorithm
  X_steam := phi*k/(k*phi + p/psat - phi);
  annotation (smoothOrder=2, Documentation(info= “<html> …);
```

(6) 構成要素の質量分率

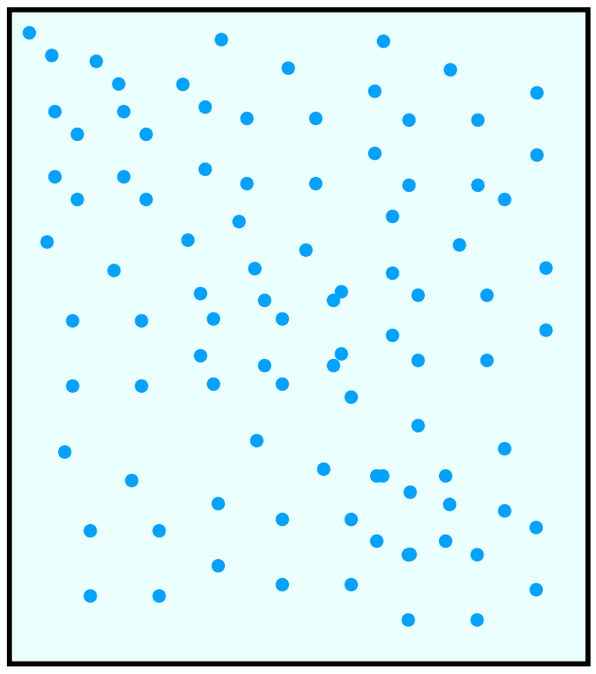
① BaseProperties の計算式

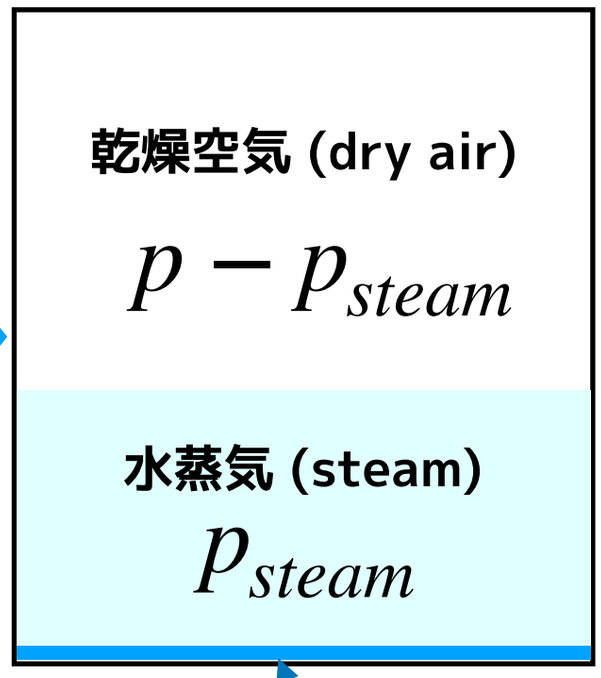
飽和水蒸気の質量分率

$$X_{liquid} = \max\left(Xi[Water] - X_{sat}, 0.0\right)$$ 液体の水、氷 (liquid, ice)

$$X_{steam} = Xi[Water] - X_{liquid}$$ 水蒸気 (steam)

$$X_{air} = 1 - Xi[Water]$$ 乾燥空気





$$X_{air} = 1 - Xi[Water]$$

X_{steam}

X_{liquid}

$Xi[Water]$

液体の水、氷 (liquid, ice)

(7) 飽和水蒸気の絶対湿度

乾燥空気 1 kg に含めることができる水蒸気の質量

飽和蒸気と乾燥空気のモル比を質量比に変換する。

① BaseProperties の計算式

飽和蒸気と乾燥空気のモル比

$$x_{sat} = k_{mair} \frac{p_{steam\_sat}}{\max(100\varepsilon, p - p_{steam\_sat})}$$

$$p_{steam\_sat} = \min(\text{saturationPressure}(T), 0.999p)$$

② MoistAir.xsaturation(State)の計算式

$$x_{sat} = k_{mair} \frac{\text{saturationPressure}(state\,.\,T)}{\max(100\varepsilon, state\,.\,p - \text{saturationPressure}(state\,.\,T))}$$

③ MoistAir.xsaturation_pT(p,_T)の計算式

$$x_{sat} = k_{mair} \frac{\text{saturationPressure}(T)}{\max(100\varepsilon, p - \text{saturationPressure}(T))}$$

(8) 相対湿度 (relative humidity)

水の分圧と水の飽和蒸気圧の比（水のモル数と飽和蒸気のモル数の比)

① BaseProperties の計算式

$$\varphi = \frac{p}{p_{steam\_sat}} \frac{Xi[Water]}{Xi[Water] + k_{mair} X_{air}} p = \frac{\dfrac{Xi[Water]}{steam.MM}}{\dfrac{Xi[Water]}{steam.MM} + \dfrac{X_{air}}{dryair.MM}} \Big/ \frac{p_{steam\_sat}}{p}$$

湿り空気中の水のモル分率

飽和蒸気のモル分率

凝縮のある状態 $\varphi > 1$ で表現できる。

② MoistAir.relativeHumidity(state)

MoistAir.relativeHumidity_pTX(p,T,X) の式

relativeHumidity(state)
= relativeHumidity_pTX(state.p, state.T, state.X)

$$\varphi = \max\left(0.0, \min\left(1.0, \frac{p}{p_{steam\_sat}} \frac{X[Water]}{X[Water] + k_{mair} X_{air}}\right)\right)$$

$0 \leq \varphi \leq 1$ になるようにリミッターがかかっている。

(9) 重量絶対湿度 (mass of total water/mass of dry air)

① BaseProperties の計算式

水 (steam + liquid + ice) の質量

$$x_{water} = \frac{Xi[Water]}{\max(X_{air}, 100\varepsilon)}$$

乾燥空気 (dry air) の質量

- 空気調和工学分野では、絶対湿度(absolute humidity) と呼ぶ。
- 気象用語では、混合比 (mixing ratio, humidity ratio) と呼ぶ。
- 単位は, [kg/kg(DA)]

(10) 気体定数 (gas constant)

① BaseProperties の計算式

$$R = dryair.R \frac{X_{air}}{1 - X_{liquid}} + steam.R \frac{X_{steam}}{1 - X_{liquid}}$$

$$= dryair.R \frac{X_{air}}{X_{air} + X_{steam}} + steam.R \frac{X_{steam}}{X_{air} + X_{steam}}$$

② MoistAir.gasConstant(state)

Return ideal gas constant as a function from thermodynamic state, only valid for phi<1

相対湿度が１未満のとき（凝縮水がないとき）のみ成立する。

$$R = dryair.R \cdot (1 - state.X[Water]) + steam.R \cdot X[Water]$$

③ MoistAir.gasConstant_X(X)

$$R = dryair.R \cdot (1 - X[Water]) + steam.R \cdot X[Water]$$

(11) 密度 (density)

① BaseProperties の計算式

$$\rho = \frac{p}{RT}$$

$$R = dryair\,.\,R\,\frac{X_{air}}{1 - X_{liquid}} + steam\,.\,R\,\frac{X_{steam}}{1 - X_{liquid}}$$

② MoistAir.density(state) の計算式

$$\rho = \frac{state\,.\,p}{\mathrm{gasConstant}(state)\;\;state\,.\,T}$$

$$dryair\,.\,R \cdot (1 - state\,.\,X[Water]) + steam\,.\,R \cdot X[Water]$$

gasConstant(state) を使うので
相対湿度が１未満のとき（凝縮水がないとき）のみ成立する。

(12) 圧力 (pressure) と温度 (temperature)

MoistAir.pressure(state)の計算式

$$p = state\,.\,p$$

MoistAir.temperature(state) の計算式

$$T = state\,.\,T$$

$$dryair\,.\,R \equiv \frac{R_{mol}}{dryair\,.\,MM}$$

$$steam\,.\,R \equiv \frac{R_{mol}}{steam\,.\,MM}$$

BaseProperties の圧力と気体成分の分圧

$$p = \rho RT = \rho \left(dryair\,.\,R\ \frac{X_{air}}{1 - X_{liquid}} + steam\,.\,R\ \frac{X_{steam}}{1 - X_{liquid}} \right) T$$

$$= \frac{1}{1 - X_{liquid}} \left(\rho X_{air}\ dryair\,.\,R\ T + \rho X_{steam}\ steam\,.\,R\ T \right)$$

$$= \boxed{\frac{1}{1 - X_{liquid}}} \left(p_{air} + p_{steam} \right)$$

$$\geq 1$$

湿り空気の圧力はモル数ベースで
計算した気体分圧の和以上になる。

$$p_{air} = \frac{\rho X_{air}}{dryair\,.\,MM} R_{mol} T, \qquad p_{steam} = \frac{\rho X_{air}}{steam\,.\,MM} R_{mol} T$$

1m3 に含まれる気体のモル数をベースに計算した分圧

(13) モル質量 (molar mass)

① BaseProperties の計算式

$$MM = \frac{1}{\frac{Xi[Water]}{MMX[Water]} + \frac{1 - Xi[Water]}{MMX[Air]}}$$

$$\Leftrightarrow \frac{1}{MM} = \frac{Xi[Water]}{MMX[Water]} + \frac{1 - Xi[Water]}{MMX[Air]}$$

湿り空気のモル数　　水のモル数　　空気のモル数

② molarMass(state) の計算式

specific gas constant[J/kg.K]

$$MM = \frac{\text{Modelica . Media . Air . MoistAir . gasConstant(state)}}{\text{Modelica . Constants . R}}$$

universal gas constant[J/mol.K]

(14) 比エンタルピ (specific enthalpy)

① BaseProperties の計算式

h = PairtialMedium.specificEnthalpy(p, T, Xi)
= MoistAir.specificEnthalpy(setState_pTX(p, T, Xi)
= MoistAir.h_pTX(state.p, state.T, state.X)

② MoistAir.h_pTX(p, T, X) の計算式

$$h = h_{steam}\ X_{steam} + h_{air}\ X_{air} + \mathrm{enthalpyOfWater}(T)\ X_{liquid}$$

(a)　(b)　(c)

MoistAir.h_pTXの algorithm より抜粋

```
h := {Modelica.Media.IdealGases.Common.Functions.h_Tlow(
      data=steam,
      T=T,
      refChoice=ReferenceEnthalpy.UserDefined,
      h_off=46479.819 + 2501014.5),
      Modelica.Media.IdealGases.Common.Functions.h_Tlow(
      data=dryair,
      T=T,
      refChoice=ReferenceEnthalpy.UserDefined,
      h_off=25104.684)}*{X_steam,X_air}
      + enthalpyOfWater(T)*X_liquid;
```

(a)水蒸気の比エンタルピ

(b)乾燥空気の比エンタルピ

(c) 液体の水、氷の比エンタルピ

(a),(b), (c) は以下の ③, ④, ⑤ も同様の式を使っている。

③ MoistAir.enthalpyOfGas の計算式

凝縮しない場合の湿り空気の比エンタルピ (空気と水蒸気)

$$h = h_{steam}\ X[Water] + h_{air}\ (1 - X[Water])$$

(a)　　(b)

MoistAir.enthalpyOfGas(T, X)

```
function extends enthalpyOfGas
    "Return specific enthalpy of gas (air and steam) as a function of temperature T and composition X"

  algorithm
    h := Modelica.Media.IdealGases.Common.Functions.h_Tlow(
          data=steam,
          T=T,
          refChoice=ReferenceEnthalpy.UserDefined,
          h_off=46479.819 + 2501014.5)*X[Water] +
      Modelica.Media.IdealGases.Common.Functions.h_Tlow(
          data=dryair,
          T=T,
          refChoice=ReferenceEnthalpy.UserDefined,
          h_off=25104.684)*(1.0 - X[Water]);
    annotation ( ...);
end enthalpyOfGas;
```

(a)水蒸気の比エンタルピ

(b)乾燥空気の比エンタルピ

④ MoistAir.enthalpyOfCondensingGas(T) の計算式

凝縮ガス（水蒸気）の比エンタルピ

algorithm

```
  h := Modelica.Media.IdealGases.Common.Functions.h_Tlow(
        data=steam,
        T=T,
        refChoice=ReferenceEnthalpy.UserDefined,
        h_off=46479.819 + 2501014.5);
```

(a)水蒸気の
比エンタルピ

⑤ MoistAir.enthalpyOfNonCondensingGas(T) の計算式

非凝縮ガス（乾燥空気）の比エンタルピ

algorithm

```
  h := Modelica.Media.IdealGases.Common.Functions.h_Tlow(
        data=dryair,
        T=T,
        refChoice=ReferenceEnthalpy.UserDefined,
        h_off=25104.684);
```

(b)乾燥空気の
比エンタルピ

(a) 水蒸気と(b)乾燥空気の比エンタルピの計算方法

Modelica.Media.IdealGases.Common.Functions. h_Tlow(data, T, exclEnthForm, refChoice, h_off)

$$h = R\frac{-a_1 + T(b_1 + a_2\log T + T(a_3 + T(\frac{1}{2}a_4 + T(\frac{1}{3}a_5 + T(\frac{1}{4}a_6 + \frac{1}{5}Ta_7)))))}{T} + h_1 + h_2$$

$$= RT\left(-\frac{a_1}{T^2} + a_2\frac{\log T}{T} + \sum_{i=1}^{7} a_i\frac{T^{i-3}}{i-2}\right) + h_1 + h_2$$

(a) data=steam, h_off=46479.819 + 2501014.5
(b) data=dryair, h_off=25104.684

$$a_i = \text{data}.\text{alow[i]},\ b_i = \text{data}.\text{blow[i]}$$

$$h_1 = \begin{cases} -\text{data}.\text{H\_f}, & \text{exclEnthForm} = \text{true} \\ 0, & \text{exclEnthForm} = \text{false} \end{cases}$$

$$h_2 = \begin{cases} 0, & \text{refChoice} = \text{ZeroAt25C} \\ \text{data}.\text{H0}, & \text{refChoice} = \text{ZeroAt0K} \\ \text{h\_off} & \text{refChoice} = \text{UserDefined}, \end{cases}$$

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 2.II

h_Tlow の algorithm の抜粋

```
algorithm
  h := data.R*((-data.alow[1] + T*(data.
    blow[1] + data.alow[2]*Math.log(T) + T*(1.*data.alow[3] + T*(0.5*data.
    alow[4] + T*(1/3*data.alow[5] + T*(0.25*data.alow[6] + 0.2*data.alow[7]*T))))))
    /T) + (if
    exclEnthForm then -data.Hf else 0.0) + (if (refChoice
     == Choices.ReferenceEnthalpy.ZeroAt0K) then data.H0 else 0.0) + (if
    refChoice == Choices.ReferenceEnthalpy.UserDefined then h_off else
          0.0);
  annotation(Inline=false,smoothOrder=2);
end h_Tlow;
```

(c) 凝縮水(液体＋固体）の比エンタルピ

① MoistAir.enthalpyOfWater(T)

$$h = \begin{cases} 4200(T - 273.15), & T \geq 273.17 \\ 2050(T - 273.15) - 333000, & T \leq 273.15 \\ \text{interpolation by spliceFunction}, & 273.15 < T < 273.17 \end{cases}$$

4200 [J/kg.K]: 水（液体の比熱）

2050[J/kg.K]: 水（固体の比熱）

33300 [J/kg]:水の融解熱

液体の比熱と固体の比熱を 273.16±0.1 [K] で滑らかにつなぐ

spliceFunction(pos, neg, x, Δx=1)

algorithm の抜粋

```
algorithm
  /*simple model assuming constant properties:
heat capacity of liquid water:4200 J/kg
heat capacity of solid water: 2050 J/kg
enthalpy of fusion (liquid=>solid): 333000 J/kg*/

  h := Utilities.spliceFunction(
        4200*(T - 273.15),
        2050*(T - 273.15) - 333000,
        T - 273.16,
        0.1);
```

② MoistAir.enthalpyOfLiquid(T)

凝縮水のエンタルピを求める別の方法。HeatCapacityOfWater(T) を積分する。

$$h = T_{degC} \times 10^3 (4.2166 - \frac{1}{2} T_{degC} (0.0033166 + \frac{1}{3} T_{degC} (0.00010295 - \frac{1}{4} T_{degC} (1.389e-6 + \frac{1}{5} T_{degC} 7.322e-9))))$$

$$h = \int cp_{fl} dT$$

algorithm の抜粋

```
algorithm
   h := (T - 273.15)*1e3*(4.2166 - 0.5*(T - 273.15)*(0.0033166 + 0.333333*(T
      - 273.15)*(0.00010295 - 0.25*(T - 273.15)*(1.3819e-6 + 0.2*(T - 273.15)
     *7.3221e-9))));
```

MoistAir.HeatCapacityOfWater(T)

$$cp_{fl} := (a_0 + a_1 T_{degC} + a_2 T_{degC}^2 + a_3 T_{degC}^3 + a_4 T_{degC}^4) \times 10^3$$

$$cp_{fl} \equiv \left(\frac{\partial h}{\partial T} \right)_p$$

$$a_0 = 4.2166,\ a_1 = -0.0033166,\ a_2 = -0.00010295,\ a_3 = 1.3819 \times 10^{-6},\ a_4 = 7.3221 \times 10^{-9}$$

The specific heat capacity of water (liquid and solid) is calculated using a polynomial approach and data from VDI-Waermeatlas 8. Edition (Db1)　**文献未確認**

algorithm の抜粋

```
algorithm
  cp_fl := 1e3*(4.2166 - (T - 273.15)*(0.0033166 + (T - 273.15)*(0.00010295
     - (T - 273.15)*(1.3819e-6 + (T - 273.15)*7.3221e-9))));
```

(15) 内部エネルギー

① BaseProperties の式

$$u = h - RT$$

② MoistAir.specificInternalEnergy(state)

$$u = \text{specificInternalEnergy\_pTX}(state.p,\ state.T,\ state.X)$$

③ MoistAir.specificInternalEnergy_pTX(p, T, X)

algorithm の抜粋

```
p_steam_sat := saturationPressure(T);
X_sat := min(p_steam_sat*k_mair/max(100*Constants.eps, p - p_steam_sat)*(
  1 - X[Water]), 1.0);
X_liquid := max(X[Water] - X_sat, 0.0);
X_steam := X[Water] - X_liquid;
X_air := 1 - X[Water];
R_gas := dryair.R*X_air/(1 - X_liquid) + steam.R*X_steam/(1 - X_liquid);
u := X_steam*Modelica.Media.IdealGases.Common.Functions.h_Tlow(
       data=steam,
       T=T,
       refChoice=ReferenceEnthalpy.UserDefined,
       h_off=46479.819 + 2501014.5) + X_air*
  Modelica.Media.IdealGases.Common.Functions.h_Tlow(
       data=dryair,
       T=T,
       refChoice=ReferenceEnthalpy.UserDefined,
       h_off=25104.684) + enthalpyOfWater(T)*X_liquid - R_gas*T;
```

(c) 液体の水、氷の比エンタルピ

(a)蒸気水の比エンタルピ

(b)乾燥空気の比エンタルピ

(d) -RT

(a),(b),(c) は比エンタルピと同様。

(16) 比エントロピ

MoistAir.specificEntropy(state)

=MoistAir.s_pTX(p,T,X)

③比エントロピの圧力依存項

空気の質量分率　　**水の質量分率**

$$s = s_{0\_Tlow}(dryair, T)(1 - X[Water]) + s_{0\_Tlow}(steam, T)X[Water] + s_p$$

①空気の比エントロピの温度依存項　　**②水の比エントロピの温度依存項**

①② 比エントロピの温度依存項
Modelica.Media.IdealGases.Common.Functions.s0_Tlow(data, T)

$$s_0(T) = R\left\{ b_2 - \frac{a_1}{2T^2} - \frac{a_2}{T} + a_3 \log T + T\left(a_4 + T\left(\frac{a_5}{2} + T\left(\frac{a_6}{3} + T \cdot \frac{a_7}{4} \right) \right) \right) \right\}$$

$$= R\left\{ -\frac{a_1}{2T^2} - \frac{a_2}{T} + a_3 \log T + \sum_{i=4}^{7} a_i \frac{T^{i-3}}{i-3} + b_2 \right\}$$

① data = dryair

② data = steam

algorithm　https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 2.II

```
s := data.R*(data.blow[2] - 0.5*data.alow[
  1]/(T*T) - data.alow[2]/T + data.alow[3]*Math.log(T) + T*(
  data.alow[4] + T*(0.5*data.alow[5] + T*(1/3*data.alow[6] + 0.25*data.alow[
  7]*T))));
```

a_i =data.alow[i]

b_2 =data.blow[2]

③比エントロピの圧力依存項

水のモル数/kg　　水のモル分率

$$s_p = -R_{mol}\left[\text{smoothMax}\left(\frac{X[Water]}{MMX[Water]}\log\left(\frac{\max\left(Y[Water],\varepsilon\right)\,p}{p_{ref}}\right),0.0,1\times10^{-9}\right)\right.$$

$$\left.+\text{smoothMax}\left(\frac{1-X[Water]}{MMX[Air]}\log\left(\frac{\max\left(Y[Air],\varepsilon\right)\,p}{p_{ref}}\right),0.0,1\times10^{-9}\right)\right]$$

空気のモル数/kg　　空気のモル分率

分圧が小さいとき下限が滑らかに 0 近づくようにリミッタをかけている。

smoothMax

bug?　https://github.com/modelica/ModelicaStandardLibrary/issues/2872

\+ になるべき！！　MSL 3.2.3

MoistAir.s_pTX(p,T,X) の algorithm

```
  s := Modelica.Media.IdealGases.Common.Functions.s0_Tlow(dryair, T)*(1 - X[
    Water]) + Modelica.Media.IdealGases.Common.Functions.s0_Tlow(steam, T)*
    X[Water] - Modelica.Constants.R*(Utilities.smoothMax(
        X[Water]/MMX[Water]*Modelica.Math.log(max(Y[Water], Modelica.Constants.eps)
      *p/reference_p),
        0.0,
        1e-9) - Utilities.smoothMax(
        (1 - X[Water])/MMX[Air]*Modelica.Math.log(max(Y[Air], Modelica.Constants.ep
      *p/reference_p),
        0.0,
        1e-9));
```

① ② ③ s_p

x1 と x 2を ±Δx の範囲で滑らかに結ぶ max 関数

MoistAir.Utilities.smoothMax(x1,x2,dx)

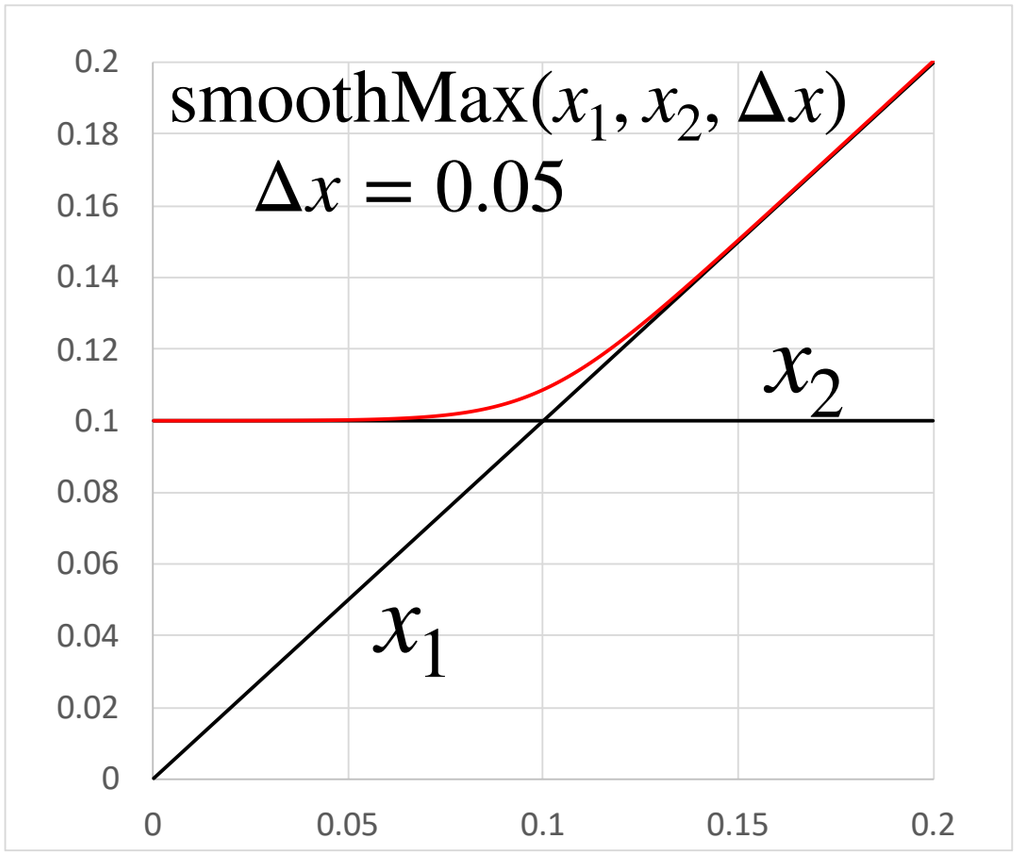
$$\mathrm{smoothMax}(x_1, x_2, \Delta x)$$

$$y = \max(x_1, x_2) + \frac{\log\left(\exp\left((4/\Delta x)\left(x_1 - \max(x_1, x_2)\right)\right) + \exp\left((4/\Delta x)\left(x_2 - \max(x_1, x_2)\right)\right)\right)}{4/\Delta x}$$

smoothMax の algorithm

```
algorithm
  y := max(x1, x2) + Math.log((exp((4/dx)*(x1 - max(x1, x2)))) + (exp((4/
    dx)*(x2 - max(x1, x2)))))/(4/dx);
  annotation (smoothOrder=2, Documentation(info="<html> ...");
end smoothMax;
```

変数 x_1 の下限を x_2 にする
リミッターとして使用する。



smoothMax(x1,x2,dx) の微分

MoistAir.Utilities.smoothMax_der(x1,x2,dx,dx1,dx2,ddx)

$$\mathrm{smoothMax\_der}(x_1, x_2, \Delta x, dx_1, dx_2, d\Delta x)$$

$x_1 > x_2$　のとき

$$dy = dx_1 + \frac{1}{4}\left\{ \frac{\left(\frac{4(dx_2 - dx_1)}{\Delta x} - \frac{4(x_2 - x_1)\ d\Delta x}{\Delta x^2}\right)\exp\left(\frac{4(x_2 - x_1)}{\Delta x}\right)}{1 + \exp\left(\frac{4(x_2 - x_1)}{\Delta x}\right)}\Delta x + \log\left(1 + \exp\left(\frac{4(x_2 - x_1)}{\Delta x}\right)\right) d\Delta x \right\}$$

$x_1 \le x_2$　のとき

$$dy = dx_2 + \frac{1}{4}\left\{ \frac{\left(\frac{4(dx_1 - dx_2)}{\Delta x} - \frac{4(x_1 - x_2)\ d\Delta x}{\Delta x^2}\right)\exp\left(\frac{4(x_1 - x_2)}{dx}\right)}{\exp\left(\frac{4(x_1 - x_2)}{\Delta x}\right) + 1}\Delta x + \log\left(\exp\left(\frac{4(x_1 - x_2)}{\Delta x}\right) + 1\right) d\Delta x \right\}$$

algorithm の抜粋

```
dy := (if x1 > x2 then dx1 else dx2) + 0.25*(((4*(dx1 - (if x1 > x2
   then dx1 else dx2))/dx - 4*(x1 - max(x1, x2))*ddx/dx^2)*exp(4*(x1 -
  max(x1, x2))/dx) + (4*(dx2 - (if x1 > x2 then dx1 else dx2))/dx - 4*(
  x2 - max(x1, x2))*ddx/dx^2)*exp(4*(x2 - max(x1, x2))/dx))*dx/(exp(4*(
  x1 - max(x1, x2))/dx) + exp(4*(x2 - max(x1, x2))/dx)) + log(exp(4*(x1
   - max(x1, x2))/dx) + exp(4*(x2 - max(x1, x2))/dx))*ddx);
```

式のなかに if~ then~else がある。場合分けして示した。

EX.3 湿り空気を等温圧縮する （volume compression）

$p = 1.01325$ [bar]

$V = 1.0$ [m^3]

$X[Air] = 0.994$

$X[Water] = 0.006$

乾燥空気
(dry air)

水蒸気
(steam)

等温圧縮

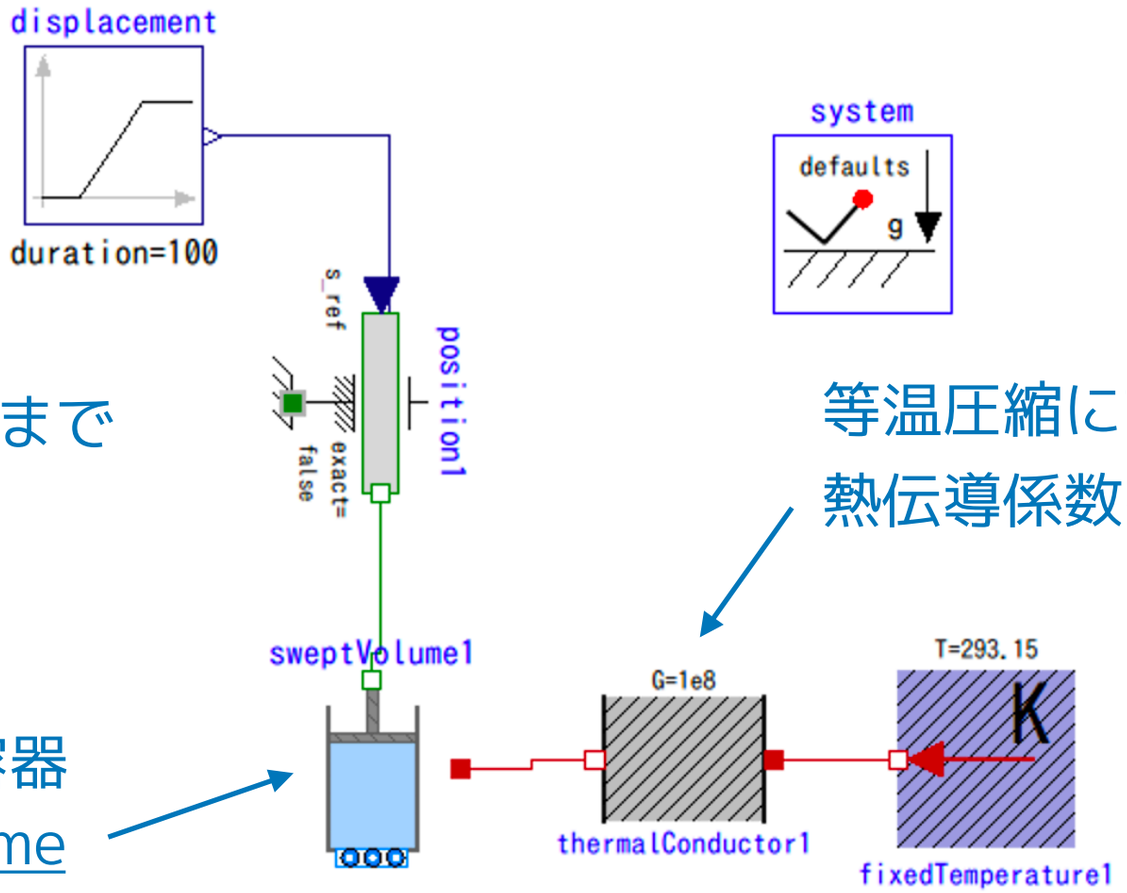
$T = 20$ [degC]

$V = 0.1$ [m^3]

乾燥空気
(dry air)

水蒸気
(steam)

液体の水
(liquid)



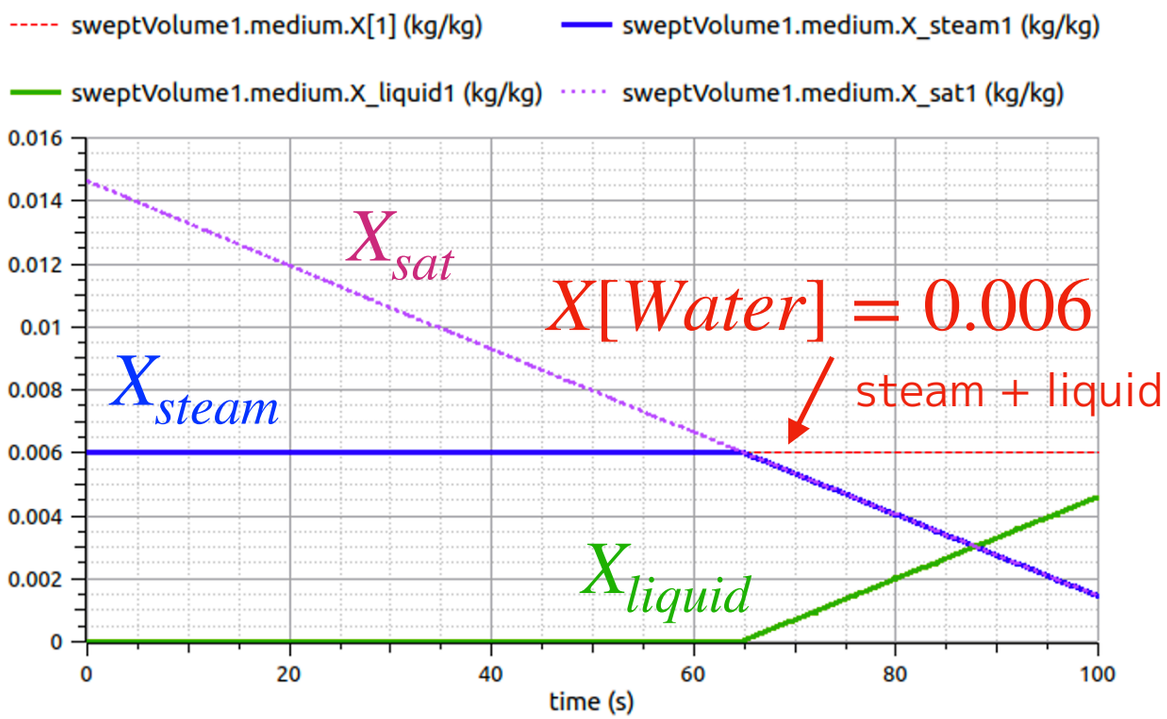
1.0 m3 から 0.1 m3 まで
圧縮する

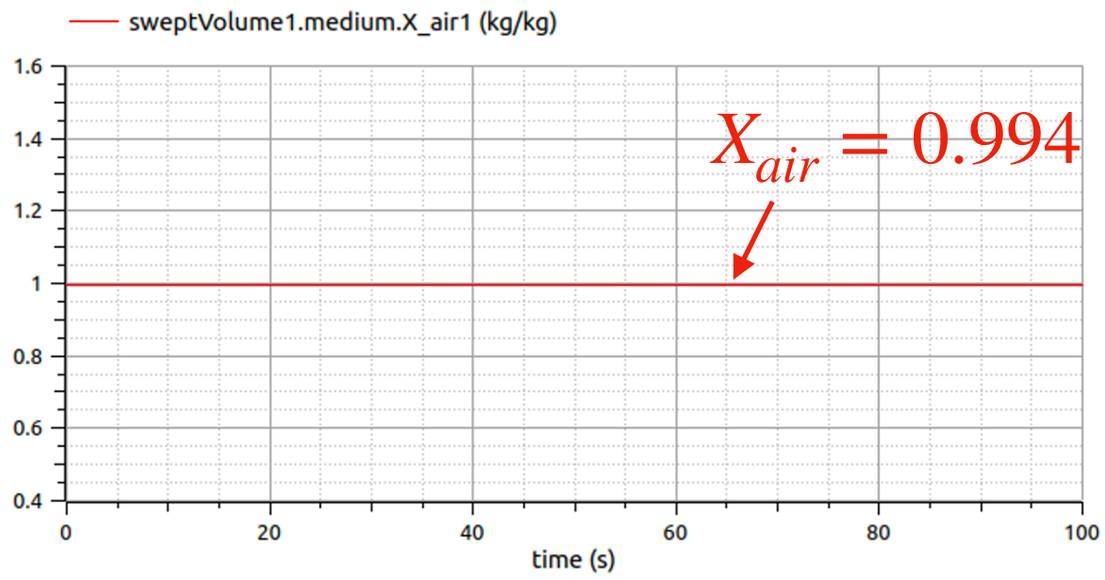
等温圧縮にするため
熱伝導係数を大きくする

容器
SweptVolume

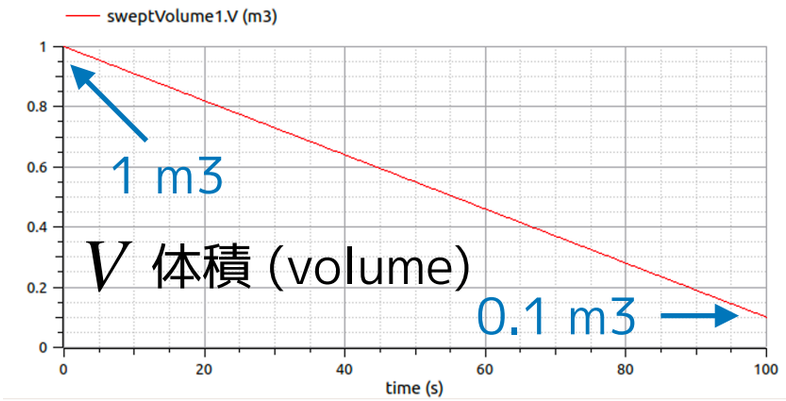
20 degC

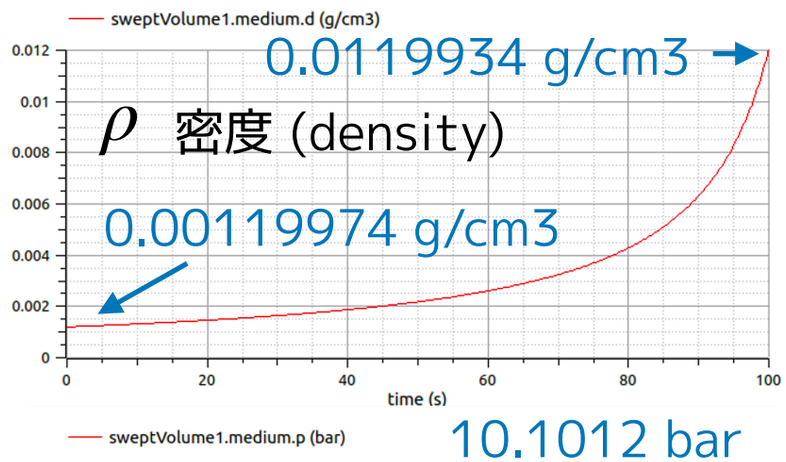
シミュレーション結果

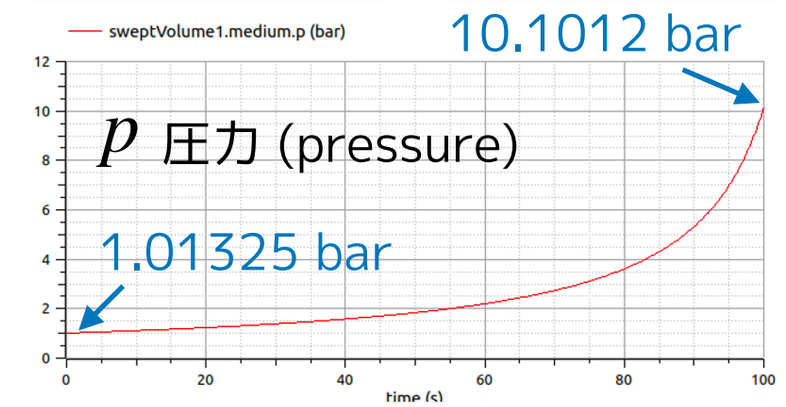


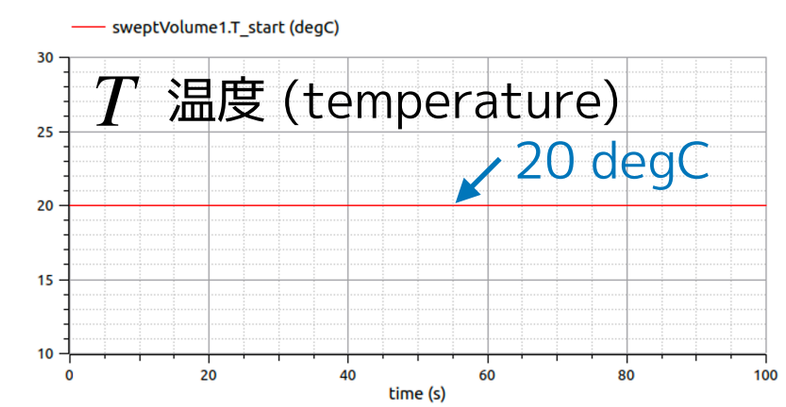


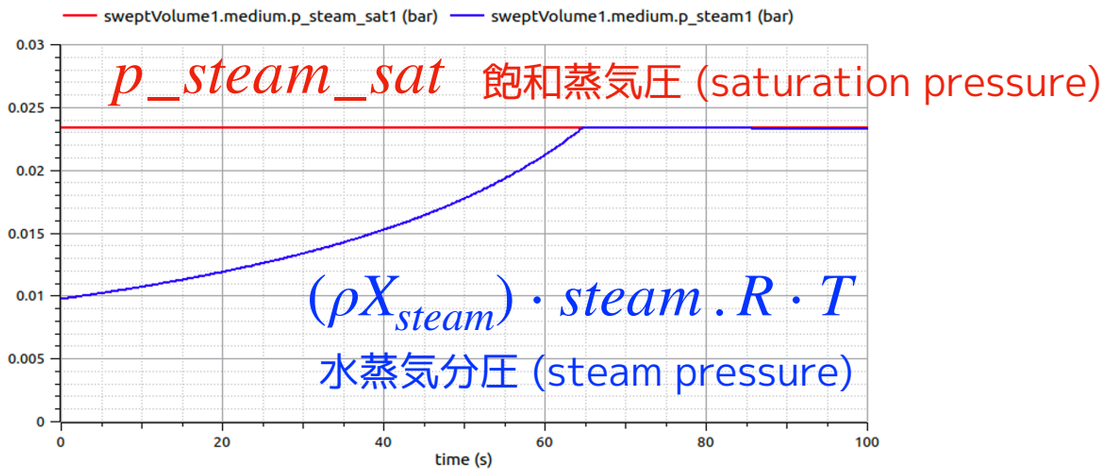
質量分率

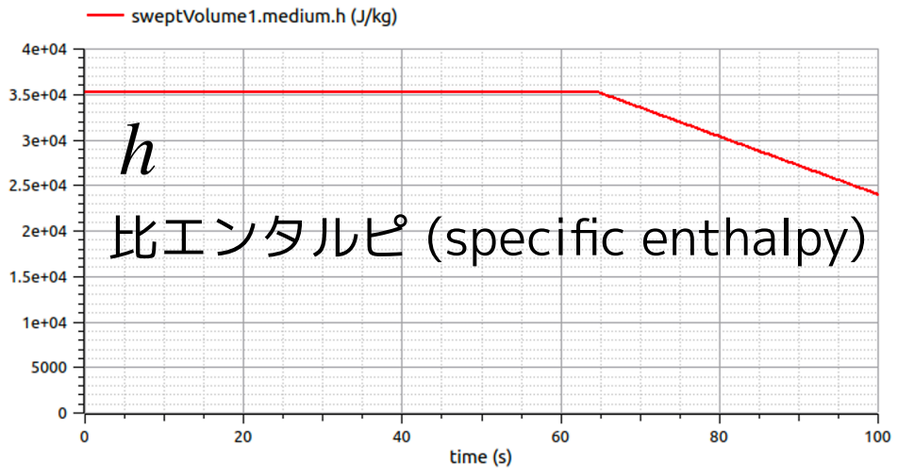


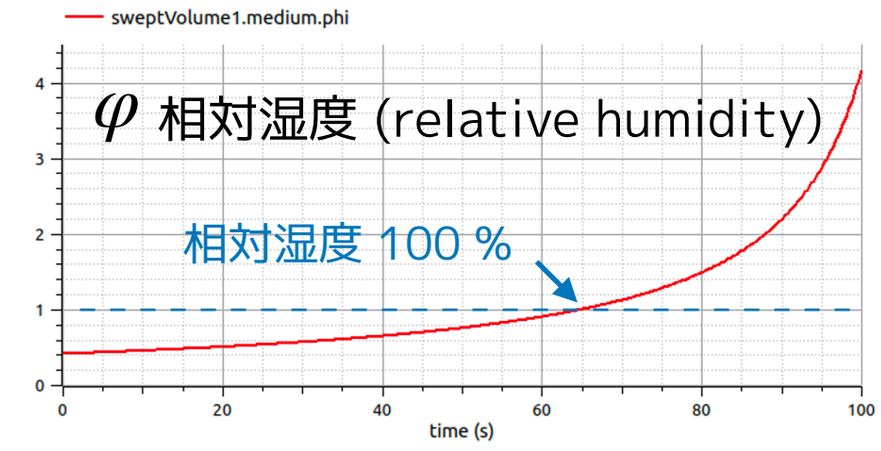


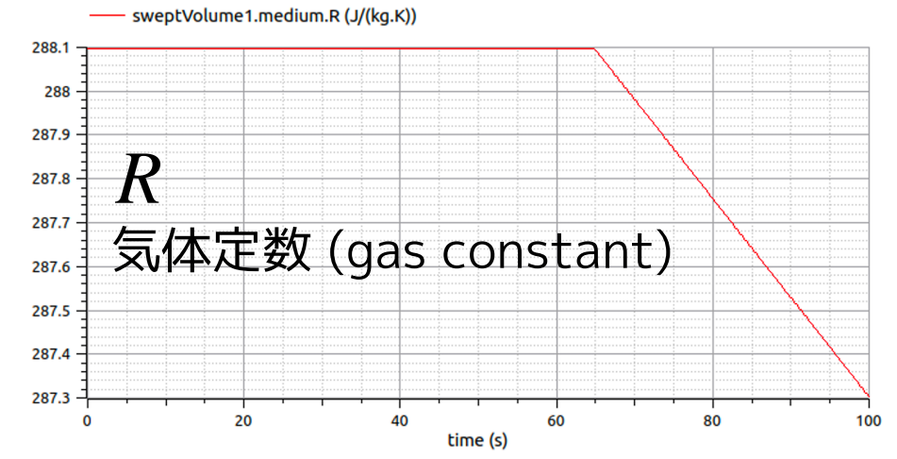


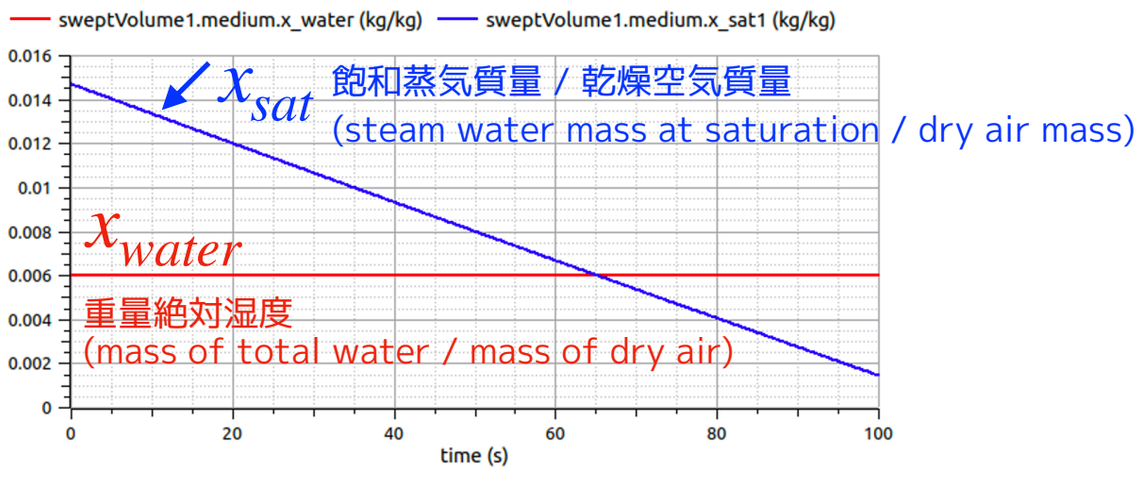


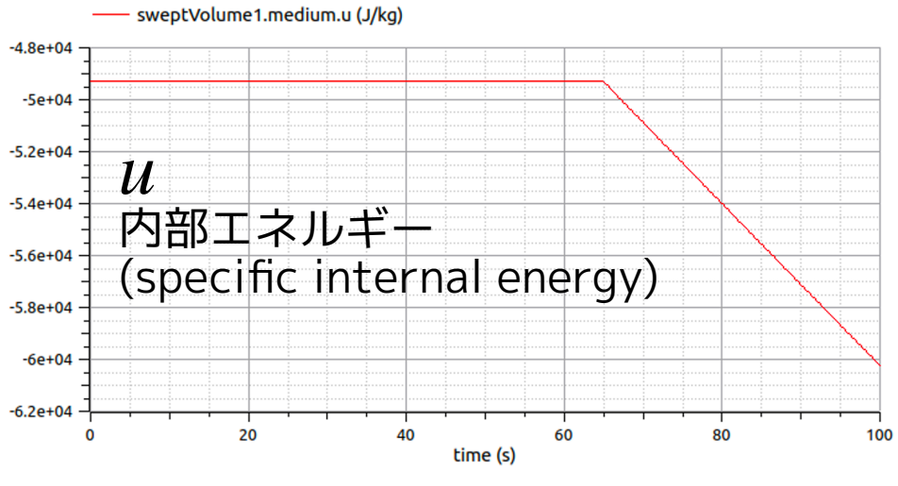

sweptVolume1.medium.p_steam_sat1 (bar)
sweptVolume1.medium.p_steam1 (bar)
p_steam_sat 飽和蒸気圧 (saturation pressure)
(ρX_steam) · steam.R · T
水蒸気分圧 (steam pressure)
time (s)


sweptVolume1.medium.h (J/kg)
h
比エンタルピ (specific enthalpy)
time (s)


sweptVolume1.medium.phi
φ 相対湿度 (relative humidity)
相対湿度 100 %
time (s)


sweptVolume1.medium.R (J/(kg.K))
R
気体定数 (gas constant)
time (s)


sweptVolume1.medium.x_water (kg/kg)
sweptVolume1.medium.x_sat1 (kg/kg)
x_sat 飽和蒸気質量 / 乾燥空気質量
(steam water mass at saturation / dry air mass)
x_water
重量絶対湿度
(mass of total water / mass of dry air)
time (s)


sweptVolume1.medium.u (J/kg)
u
内部エネルギー
(specific internal energy)
time (s)

VolumeCompression

```
model VolumeCompression
  replaceable package Medium = MyMoistAir;
   Modelica.Fluid.Machines.SweptVolume sweptVolume1(redeclare package Medium = Medium, T_start = 293.15,
    X_start = {0.006, 0.994}, clearance = 0.1,
    energyDynamics = Modelica.Fluid.Types.Dynamics.SteadyStateInitial,
    massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    p_start = 101325, pistonCrossArea = 1,
    use_HeatTransfer = true,
    use_T_start = true)  annotation( …);
   Modelica.Mechanics.Translational.Sources.Position position1(v(fixed = true, start = 0))  annotation( …);
   Modelica.Blocks.Sources.Ramp displacement(duration = 100, height = -0.9, offset = 0.9, startTime = 0)  annotation( …);
   Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature1(T = 293.15)  annotation( …);
   inner Modelica.Fluid.System system annotation( …);
   Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 1e8)  annotation( …);
equation
   connect(sweptVolume1.heatPort, thermalConductor1.port_b) annotation( …);
   connect(thermalConductor1.port_a, fixedTemperature1.port) annotation( …);
   connect(position1.flange, sweptVolume1.flange) annotation( …);
   connect(displacement.y, position1.s_ref) annotation( …);
   annotation( …);
end VolumeCompression;
```

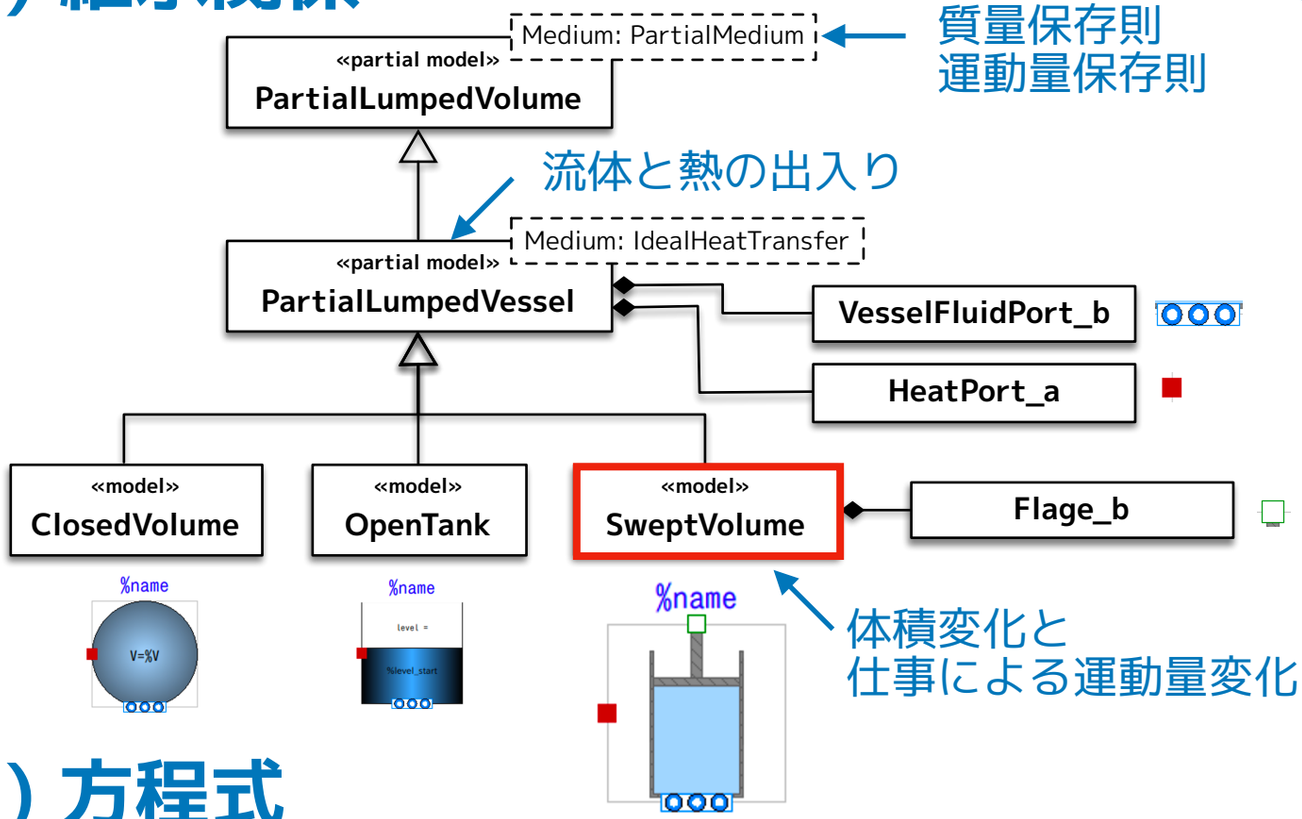
MyMoistAir

```
package MyMoistAir
    extends Modelica.Media.Air.MoistAir;

    model extends BaseProperties
            MassFraction X_liquid1 "Mass fraction of liquid or solid water";
            MassFraction X_steam1 "Mass fraction of steam water";
            MassFraction X_air1 "Mass fraction of air";
            MassFraction X_sat1 "Steam water mass fraction of saturation boundary in kg_water/kg_moistair";
            MassFraction x_sat1 "Steam water mass content of saturation boundary in kg_water/kg_dryair";
            AbsolutePressure p_steam_sat1 "partial saturation pressure of steam";
            AbsolutePressure p_steam1;
            AbsolutePressure p_air1;
            Real p_diff;

        equation
            X_liquid1 = X_liquid;
            X_steam1 = X_steam;
            X_air1 = X_air;
            X_sat1 = X_sat;
            x_sat1 = x_sat;
            p_steam_sat1 = p_steam_sat;
            p_steam1 = d * X_steam * steam.R * T;
            p_air1 = d * X_air * dryair.R * T;
            p_diff = p - (p_air1 + p_steam1);
    end BaseProperties;
equation

end MyMoistAir;
```

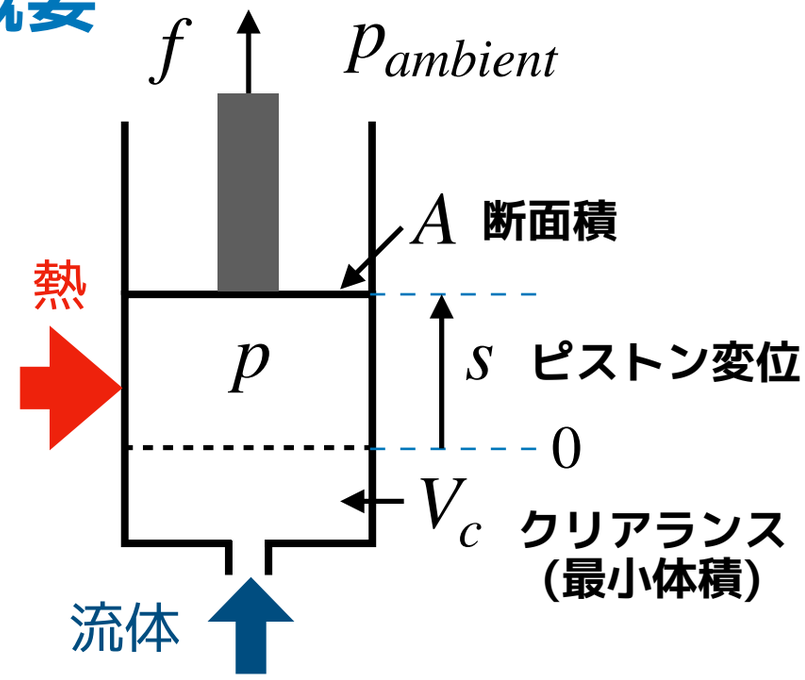
MoistAir を継承した新たな
パッケージをつくり、
新たな変数を作って、
protected variables や
値を確認したい関数の結果
などを代入して、出力できるようにする。

Modelica.Fluid.Machines.SweptVolume

(1) 継承関係



(2) 概要



(3) 方程式

$$V = V_c + sA$$ 体積

$$0 = f + (p - p_{ambient})A$$ 力のバランス

$$Wb_{flow} = pA\left(-\frac{ds}{dt}\right)$$ 仕事による運動量変化

$$p = ps_i$$ 容器内圧力とVesselFluidPort の静圧が等しい

s = flange.s

f = flange.f

V_c = clearance

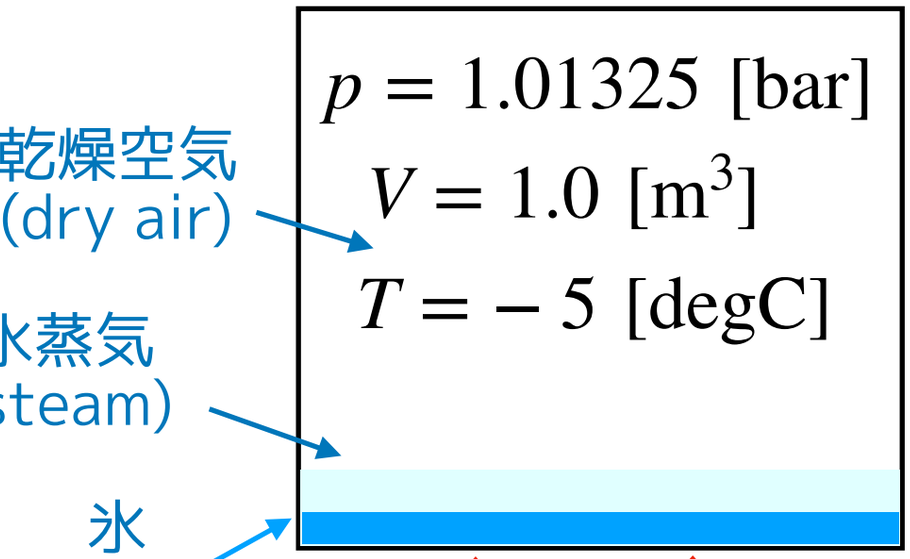
A = pistionCrossArea

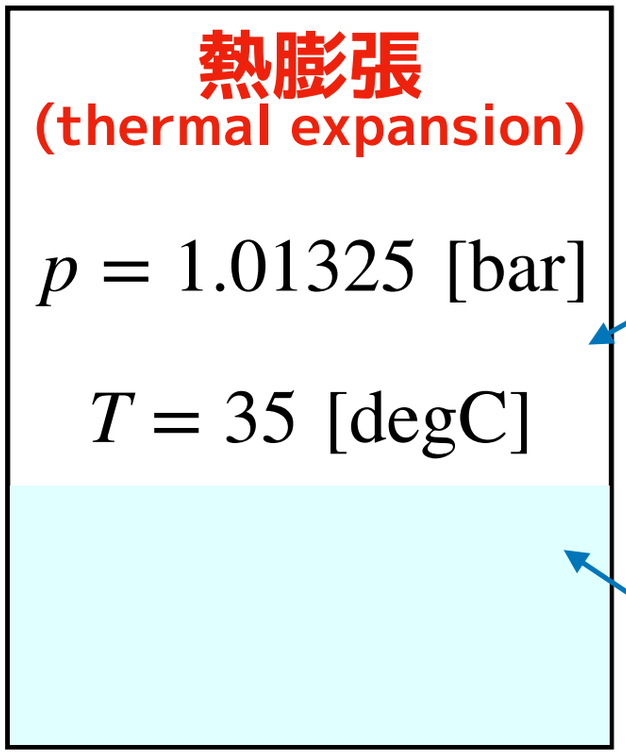
p = medium.p

$p_{ambient}$ = system.p_ambient

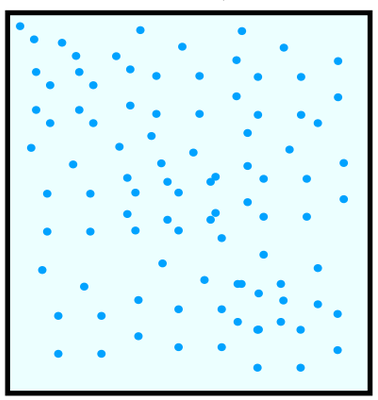
ps_i = vessel_ps_static[i]

EX.4 湿り空気を温める (volume heating)





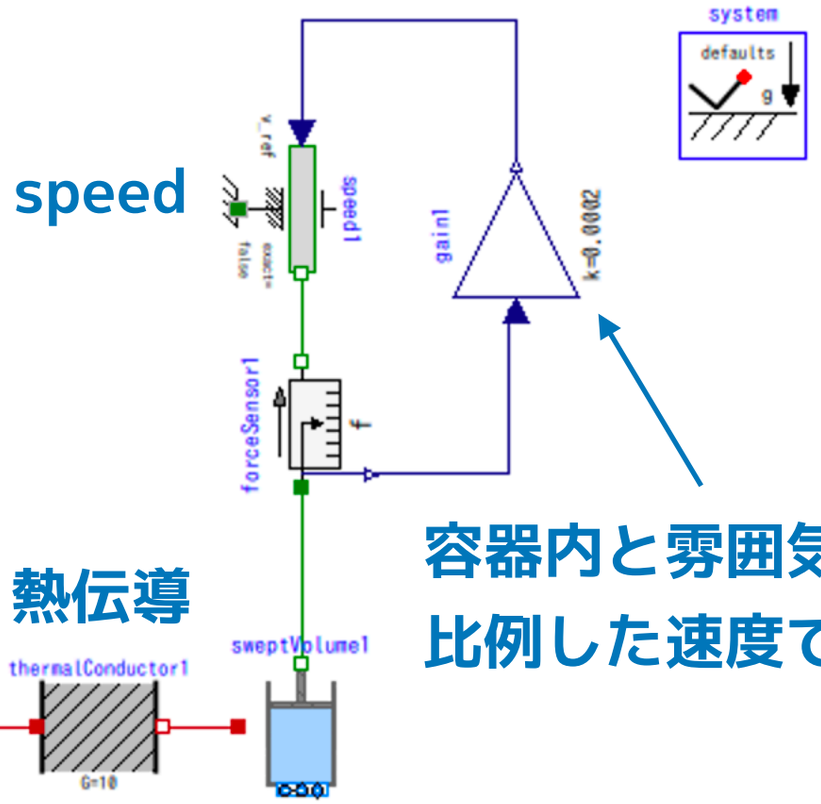
熱伝導による加熱



霧 (fog)

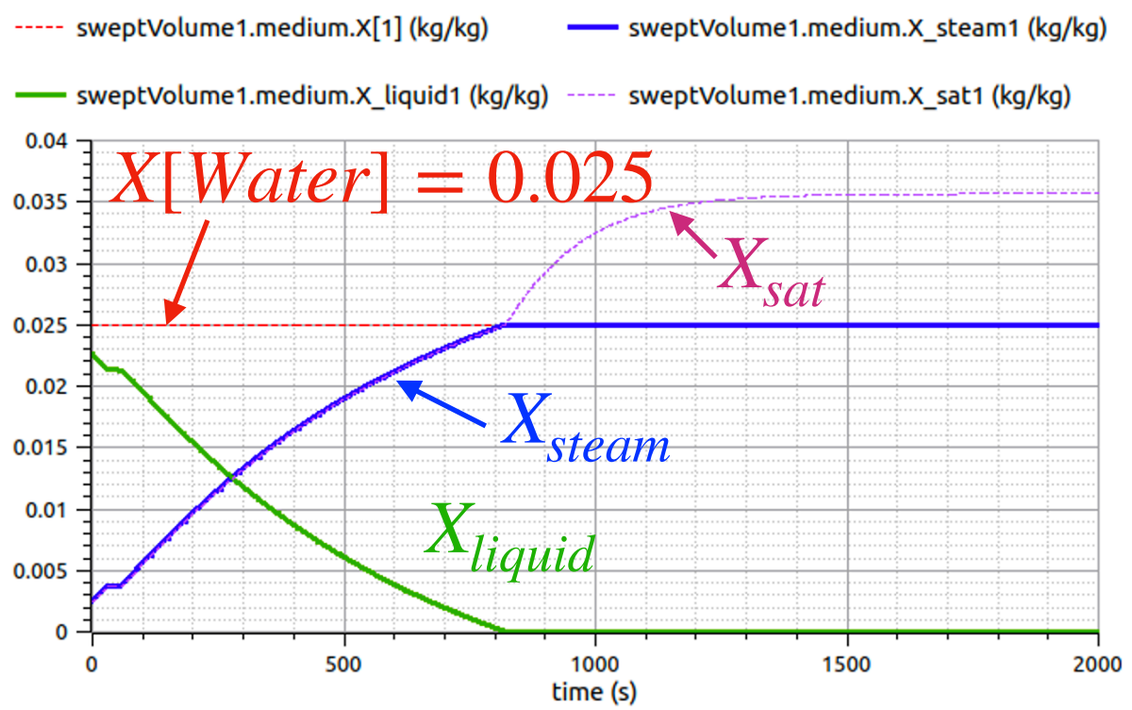
質量比2.5%の水分

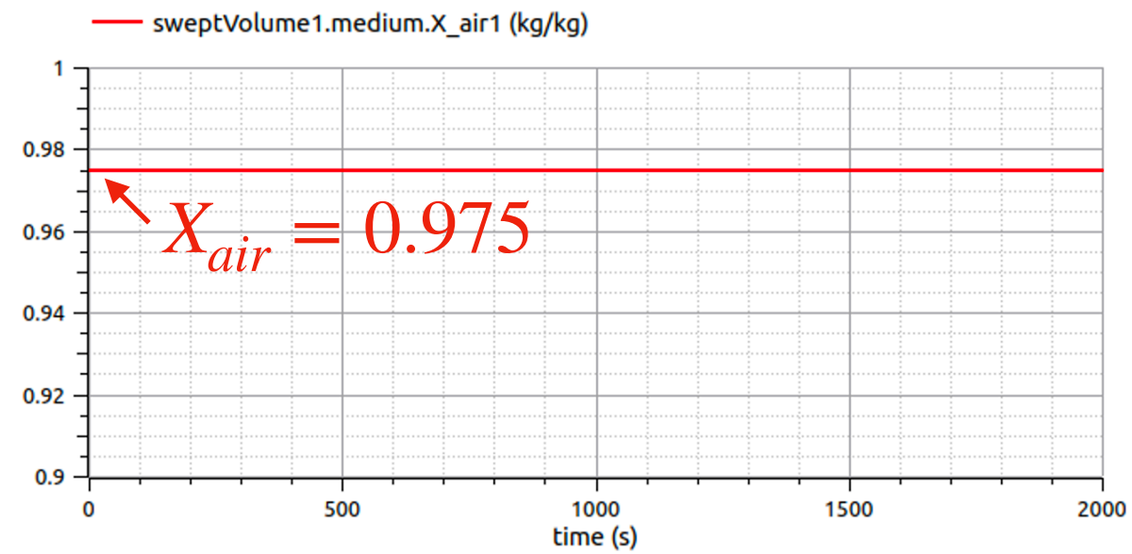
$$X[Air] = 0.975$$
$$X[Water] = 0.025$$



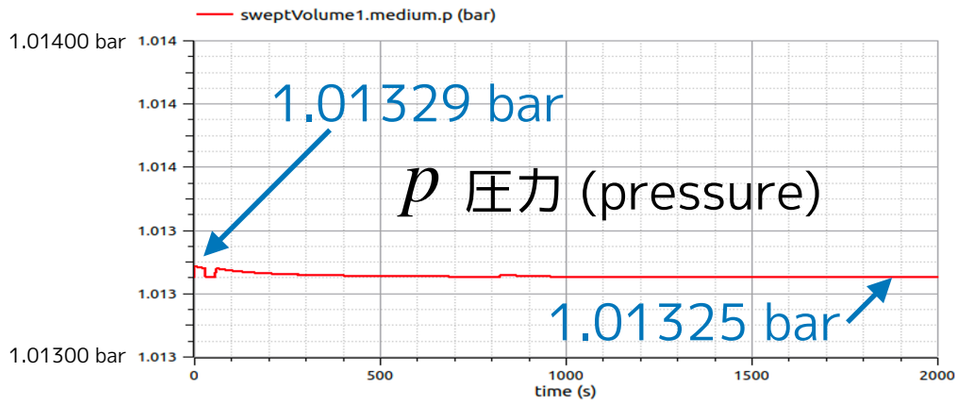
容器内と雰囲気の圧力差に比例した速度で膨張させる

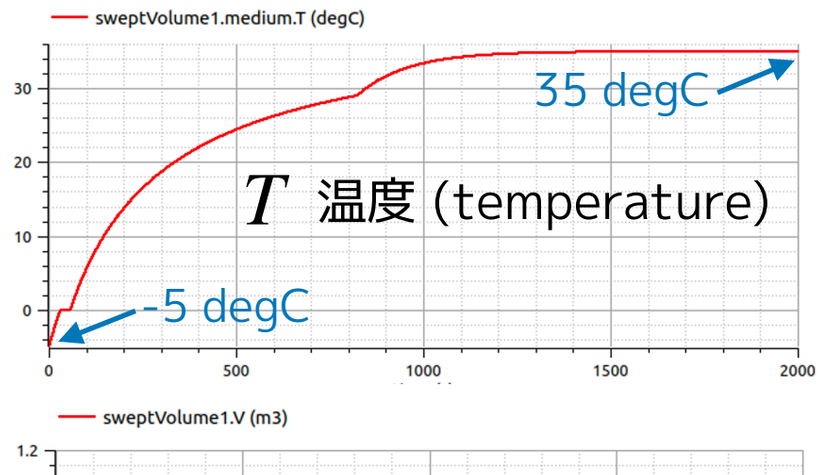
シミュレーション結果

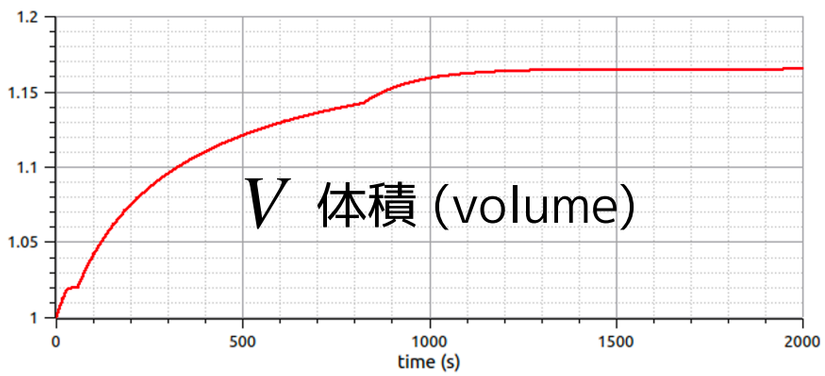


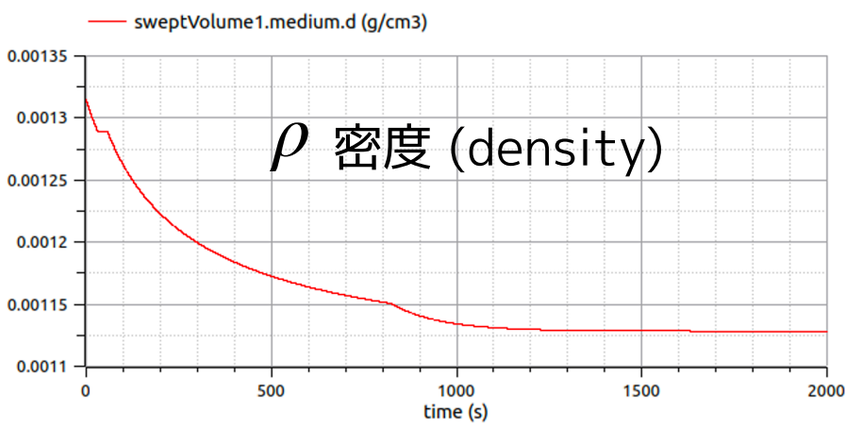


質量分率

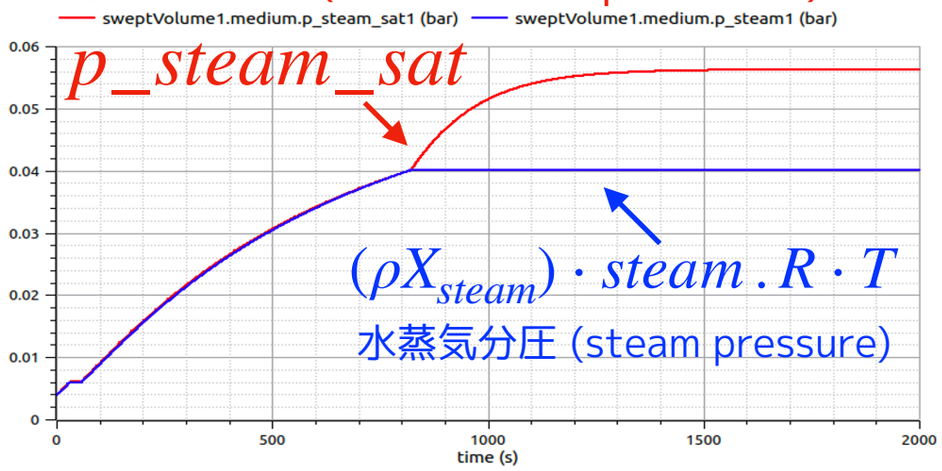


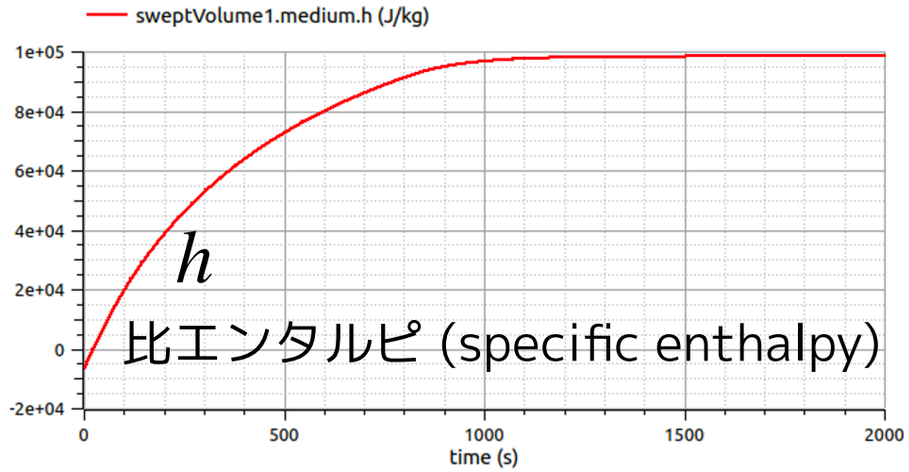


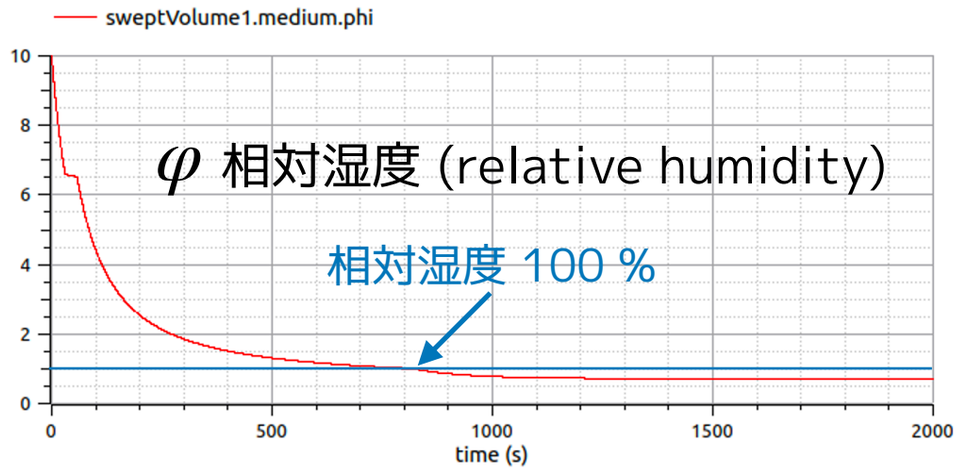


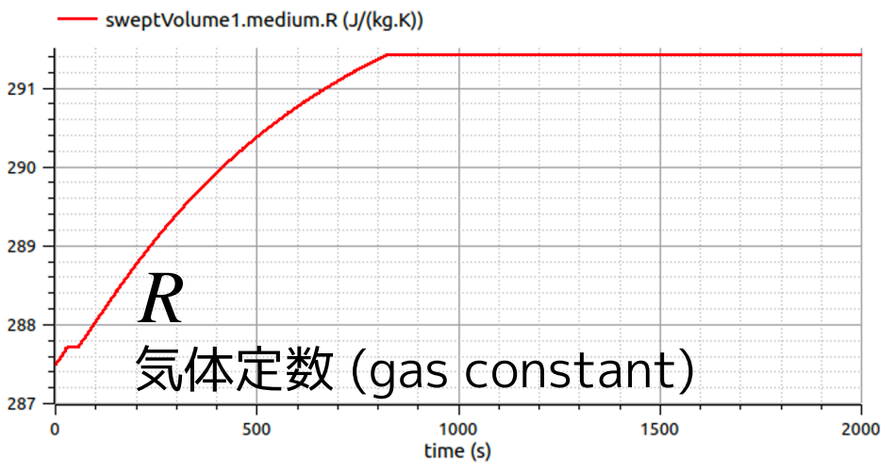


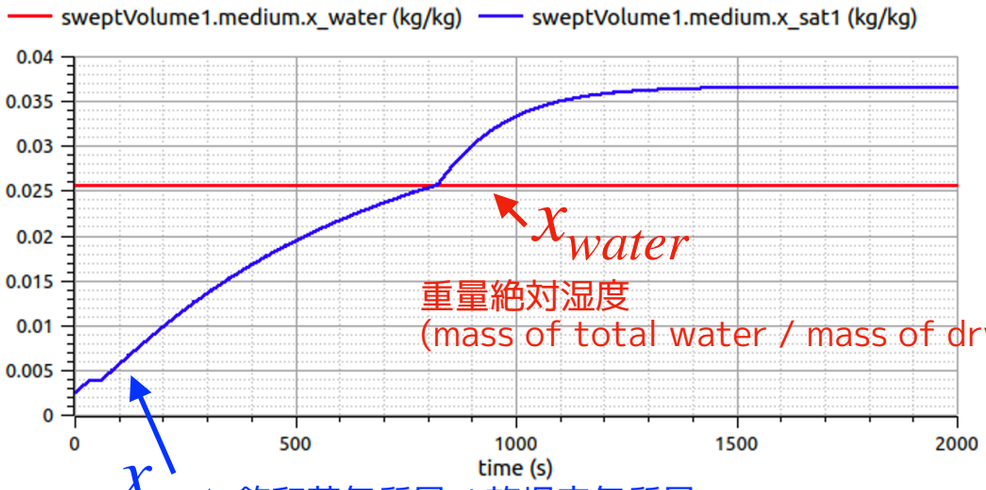
飽和蒸気圧 (saturation pressure)

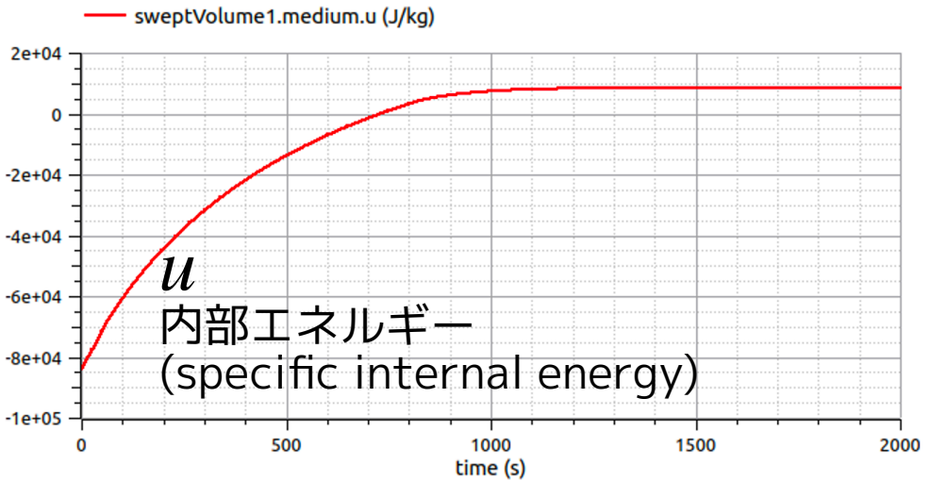












VolumeHeating

```
model VolumeHeating
   replaceable package Medium = MyMoistAir;
   inner Modelica.Fluid.System system annotation( …);
   Modelica.Fluid.Machines.SweptVolume sweptVolume1(redeclare package Medium = Medium,
     T_start = 268.15, X_start = {0.025, 0.975}, clearance = 0.1,
     energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
     massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
     nPorts = 0, p_start = 101325, pistonCrossArea = 1, use_HeatTransfer = true,
     use_T_start = true) annotation( …);
   Modelica.Mechanics.Translational.Sensors.ForceSensor forceSensor1 annotation( …);
   Modelica.Mechanics.Translational.Sources.Speed speed1(s(start = 0.9))  annotation( …);
   Modelica.Blocks.Math.Gain gain1(k = 0.0002) annotation( …);
   Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 10)  annotation( …);
   Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature1(T = 308.15)  annotation( …);
equation
   connect(gain1.y, speed1.v_ref) annotation( …);
   connect(forceSensor1.f, gain1.u) annotation( …);
   connect(sweptVolume1.heatPort, thermalConductor1.port_b) annotation( …);
   connect(forceSensor1.flange_a, sweptVolume1.flange) annotation( …);
   connect(thermalConductor1.port_a, fixedTemperature1.port) annotation( …);
   connect(speed1.flange, forceSensor1.flange_b) annotation( …);
   annotation( …);
end VolumeHeating;
```

5. setState_XXX 関数

熱力学的状態変数の組み合わせから ThemodynamicState レコードを返す関数

MoistAir.setState_pTX(p, T, X)
algorithm の抜粋

```
algorithm
  state := if size(X, 1) == nX then ThermodynamicState(
        p=p,
        T=T,
        X=X) else ThermodynamicState(
        p=p,
        T=T,
        X=cat(
          1,
          X,
          {1 - sum(X)}));
```

質量分率
$\mathbf{X}$ の代わりに $\mathbf{X}_i$
を使うことができる。

MoistAir.setState_dTX(d, T, X)
algorithm の抜粋

```
algorithm
  state := if size(X, 1) == nX then ThermodynamicState(
        p=d*({steam.R,dryair.R}*X)*T,
        T=T,
        X=X) else ThermodynamicState(
        p=d*({steam.R,dryair.R}*cat(
          1,
          X,
          {1 - sum(X)}))*T,
      T=T,
      X=cat(
        1,
        X,
        {1 - sum(X)}));
```

密度と温度から圧力を計算する

$$p = \frac{1}{v}RT = \rho RT$$

$$R = \text{steam.R}\ \text{X[Water]} + \text{dryair.R}\ \text{X[Air]}$$
$$= \{\text{steam.R}, \text{dryair.R}\} * \text{X}$$

凝縮があるとき $(\varphi > 1)$ は
R の計算方法が異なるので
BasePropertiesの圧力と一致しない

MoistAir.setState_phX(p, h, X)

algorithm (抜粋)

```
algorithm
  state := if size(X, 1) == nX then ThermodynamicState(
        p=p,
        T=T_phX(    ①
          p,
          h,
          X),
        X=X) else ThermodynamicState(
        p=p,
        T=T_phX(    ①
          p,
          h,
          X),
        X=cat(
          1,
          X,
          {1 - sum(X)}));
```

MoistAir.setState_psX(p,s,X)

algorithm (抜粋)

```
algorithm
  state := if size(X, 1) == nX then ThermodynamicState(
        p=p,
        T=T_psX(    ②
          p,
          s,
          X),
        X=X) else ThermodynamicState(
        p=p,
        T=T_psX(    ②
          p,
          s,
          X),
        X=cat(
          1,
          X,
          {1 - sum(X)}));
```

温度の計算に

① **MoistAir.T_phX(p, h, X)**

② **MoistAir.T_psX(p, s, X)**

を使用する。

① T_phX(p, h, X) と ② T_psX(p, s, X)

非線形方程式ソルバーパッケージ

Modelica.Media.Common.OneNonLinearEquation

を継承したパッケージ Internal 使用して温度 T を計算する。

MoistAir.T_phX.Internal.f_nonlinear(x,p,X, f_nonlinear_data)

```
redeclare function extends f_nonlinear
algorithm
  y := h_pTX(
            p,
            x,
            X);
end f_nonlinear;
```

非線形方程式

$$y = \mathrm{h\_pTX}(p, x, X)$$

比エンタルピ　h_pTX(p,T,X)

MoistAir.T_phX.Internal.f_nonlinear(x,p,X, f_nonlinear_data)

```
redeclare function extends f_nonlinear
algorithm
  y := s_pTX(
            p,
            x,
            X);
end f_nonlinear;
```

非線形方程式

$$y = \mathrm{s\_pTX}(p, x, X)$$

比エントロピ　s_pTX(p,T,X)

非線形方程式の解法の詳細

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 5.II 温度を求める関数

6. その他の物性関数

(1) 飽和温度 (saturation temperature)

MoistAir.saturationspeTemperature(p, T_min=190, T_max=647)

非線形方程式ソルバーパッケージ

Modelica.Media.Common.OneNonLinearEquation

を継承したパッケージ Internal 使用して温度 T を計算する。

非線形方程式

$$y = \text{saturationPressure}(x)$$

nonlinear equation　　**飽和蒸気圧**　**MoistAir.satureationPressure(T)**

```
redeclare function extends f_nonlinear
algorithm
  y := saturationPressure(x);
  // Compute the non-linear equation: y = f(x, Data)
end f_nonlinear;
```

非線形方程式の解法の詳細は以下を参照してください。

https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 5.II 温度を求める関数

(2) 定圧比熱 (specific heat capacity Cp)

MoistAir.specificHeatCapacityCp(state) の計算式

$$h = \mathrm{h\_pTX}(p, T, \mathbf{X})$$

$$dh = \mathrm{h\_pTX\_der}(p, T, \mathbf{X}, dp, dT, d\mathbf{X}) = \left(\frac{\partial h}{\partial p}\right) dp + \left(\frac{\partial h}{\partial T}\right) dT + \sum \left(\frac{\partial h}{\partial Xi}\right) dXi$$

(a) 比エンタルピの全微分

$$\Rightarrow c_p(state) = \mathrm{h\_ptX\_der}(state.p, state.T, state.X, dp = 0, dT = 1, d\mathbf{X} = \mathbf{0})$$

$$= \left(\frac{\partial h}{\partial T}\right)_p$$

algorithm (抜粋)

```
function extends specificHeatCapacityCp
    "Return specific heat capacity at constant pressure as a function of the thermodynamic state record"

  protected
    Real dT(unit="s/K") = 1.0;
  algorithm
    cp := h_pTX_der(
          state.p,
          state.T,
          state.X,
          0.0,
          1.0,
          zeros(size(state.X, 1)))*dT "Definition of cp: dh/dT @ constant p" ; ⋯
```

$$c_p = \mathrm{h\_pTX\_der}(state.p, state.T, state.\mathbf{X}, dp = 0, dT = 1, d\mathbf{X} = \mathbf{0})$$

(a)

(a) 比エンタルピの全微分

MoistAir.h_pTX_der(p, T, X, dp, dT, dX)

$$h = \overset{(e)}{X_{steam}} h_{steam} + \overset{(i)}{X_{air}} h_{air} + \overset{(m)}{X_{liquid}} h_{liquid}$$

比エンタルピ h_pTX(p, T, X)

$$dh = X_{steam} dh_{steam} + dX_{steam} h_{steam} + X_{air} dh_{air} + dX_{air} h_{air} + X_{liquid} dh_{liquid} + dX_{liq} h_{liquid}$$

(b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m)

h_pTX_der の algorithm の抜粋1

```
h_der := X_steam*Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(
    data=steam,
    T=T,
    refChoice=ReferenceEnthalpy.UserDefined,
    h_off=46479.819 + 2501014.5,
    dT=dT) + dX_steam*Modelica.Media.IdealGases.Common.Functions.h_Tlow(
    data=steam,
    T=T,
    refChoice=ReferenceEnthalpy.UserDefined,
    h_off=46479.819 + 2501014.5) + X_air*
 Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(
    data=dryair,
    T=T,
    refChoice=ReferenceEnthalpy.UserDefined,
    h_off=25104.684,
    dT=dT) + dX_air*Modelica.Media.IdealGases.Common.Functions.h_Tlow(
    data=dryair,
    T=T,
    refChoice=ReferenceEnthalpy.UserDefined,
    h_off=25104.684) + X_liquid*enthalpyOfWater_der(T=T, dT=dT) +
 dX_liq*enthalpyOfWater(T);
```

(b)(c)　(d)(e)　(f)(g)　(h)(i)　(j)(k)　(l)(M)

$$p_{steam\_sat} = \text{saturationPressure}(T)$$

$$x_{sat} = k_{mair}\frac{p_{steam\_sat}}{\max(100\varepsilon, p - p_{steam\_sat}}$$

$$X_{sat} = \min(x_{sat}(1 - X[Water]),1.0)$$

(j) $$X_{liquid} = \text{smoothMax}(X[Water] - X_{sat},0.0,0.05)$$

(b) $$X_{steam} = X[Water] - X_{liquid}$$

(f) $$X_{air} = 1 - X[Water]$$

(h) $$dX_{air} = -dX[Water]$$

$$dps = \text{saturationPressure\_der}(T_{sat} = T, dT_{sat} = dT)$$

$$dX_{sat} = k_{mair}\left\{\frac{d(p_{steam\_sat})}{p - p_{steam\_sat}} - \frac{p_{steam\_sat}}{(p - p_{steam\_sat})^2}(dp - d(p_{steam\_sat}))\right\}$$

$$= k_{mair}\frac{d(p_{steam\_sat})\ (p - p_{steam\_sat}) - p_{steam\_sat}(dp - d(p_{steam\_sat}))}{(p - p_{steam\_sat})^2}$$

$$= k_{mair}\frac{dps\ (p - p_{steam\_sat}) - p_{steam\_sat}(dp - dps))}{(p - p_{steam\_sat})^2}$$

(l) $$dX_{liq} = \text{Utilities.smoothMax\_der}(X[Water] - X_{sat},\ 0.0,\ 1.0 \times 10^{-5},\ (1 + x_{sat})dX[Water] - (1 - X[Water])dx_{sat},\ 0,0)$$

$$X_{liquid} = X[Water] - X_{sat} = X[Water] - x_{sat}(1 - X[Water])$$

$$dX_{liquid} = dX[Water] - (dx_{sat}(1 - X[Water]) + x_{sat}(-dX[Water]))$$

$$= (1 + x_{sat})dX[Water] - dx_{sat}(1 - X[Water])$$

(d) $$dX_{steam} = dX[Water] - dX_{liq}$$

h_pTX_der の algorithm の抜粋2

```
p_steam_sat := saturationPressure(T);
x_sat := p_steam_sat*k_mair/max(100*Modelica.Constants.eps, p - p_steam_sat);
X_sat := min(x_sat*(1 - X[Water]), 1.0);
X_liquid := Utilities.smoothMax(
      X[Water] - X_sat,
      0.0,
      1e-5);                                                          (j)
X_steam := X[Water] - X_liquid;
X_air := 1 - X[Water];                                                (b)

dX_air := -dX[Water];                                                 (f)
dps := saturationPressure_der(Tsat=T, dTsat=dT);                       (h)
dx_sat := k_mair*(dps*(p - p_steam_sat) - p_steam_sat*(dp - dps))/(p -
  p_steam_sat)/(p - p_steam_sat);
dX_liq := Utilities.smoothMax_der(
      X[Water] - X_sat,
      0.0,
      1e-5,                                                           (l)
      (1 + x_sat)*dX[Water] - (1 - X[Water])*dx_sat,
      0,
      0);
dX_steam := dX[Water] - dX_liq;                                       (d)
```

(c) 水蒸気の比エンタルピの温度微分）　dh_{stream}

h_pTX_der の algorithm の抜粋3

```
Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(
    data=steam,
    T=T,
    refChoice=ReferenceEnthalpy.UserDefined,
    h_off=46479.819 + 2501014.5,
    dT=dT)
```

標準状態 (25 degC, 101325 Pa) の比エンタルピ

46479.819 + 2501014.5 J/kg.K

(g) 乾燥空気の比エンタルピの温度微分　dh_{air}

h_pTX_der の algorithm の抜粋4

```
Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(
    data=dryair,
    T=T,
    refChoice=ReferenceEnthalpy.UserDefined,
    h_off=25104.684,
    dT=dT)
```

標準状態 (25 degC, 101325 Pa) の比エンタルピ

25104.684 J/kg.K

低温領域の理想気体の比エンタルピ

h_Tlow_der(data, T, exclEnthForm, refChohice, h_off, dt)

Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(data, T, exclEnthForm, refChohice, h_off, dt)

比エンタルピーの微分なので
積分定数に相当する h_off は引数から消える。

$$\text{h\_der} = dT\ \text{cp\_Tlow}(data, T), \quad data = steam, dryair$$

(c)　(g)

algorithm の抜粋

```
h_der := dT*Modelica.Media.IdealGases.Common.Functions.cp_Tlow(
                    data,T);
annotation(Inline=true,smoothOrder=2);
```

Modelica.Media.IdealGAses.Common.Functions.cp_Tlow(data, T)

$$\text{cp\_Tlow}(data, T) = R\left\{\frac{1}{T^2}(a_1 + T(a_2 + T(a_3 + T(a_4 + T(a_5 + T(a_6 + a_7 T)))))) \right\}$$

$$T \le T_{limit} \qquad = R\sum_{i=1}^{7} a_i T^{i-3}, \qquad a_i = \text{data}\,.\,\text{a\_low}[i],\ i = 1,...,7$$

の比熱　https://www.amane.to/wp-content/uploads/2018/10/Introduction_SingleGasNasa_20181007.pdf 2.II

algorithm の抜粋

```
cp := data.R*(1/(T*T)*(data.alow[1] + T*(
  data.alow[2] + T*(1.*data.alow[3] + T*(data.alow[4] + T*(data.alow[5] + T
  *(data.alow[6] + data.alow[7]*T)))))));
annotation (Inline=false, derivative(zeroDerivative=data) = cp_Tlow_der);
```

(k) 液体または固体の水の比エンタルピの微分

MoistAir.enthalpyOfWater_der(Tsat, dTsat)

algorithm の抜粋

```
  dh := Utilities.spliceFunction_der(
        4200*(T - 273.15),
        2050*(T - 273.15) - 333000,
        T - 273.16,
        0.1,
        4200*dT,
        2050*dT,
        dT,
        0);
```

4200 [J/kg.K]: 水（液体の比熱）

2050[J/kg.K]: 水（固体の比熱）

33300 [J/kg]:水の融解熱

$$h = \begin{cases} 4200(T - 273.15), & T \geq 273.17 \\ 2050(T - 273.15) - 333000, & T \leq 273.15 \end{cases}$$

$$dh = \begin{cases} 4200\ dT \\ 2050\ dT \end{cases}$$

に spliceFunction の微分 spliceFunction_der を適用する。

(3) 内部エネルギーの微分 (derivative of specific internal energy)

MoistAir.specificHeatCapacityCp(state) の計算式

$$u = h - RT = \left\{ X_{steam} h_{steam} + X_{air} h_{air} + X_{liquid} h_{liquid} \right\} - RT \quad \text{より}$$

(b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m)

$$du = X_{steam} dh_{steam} + dX_{steam} h_{steam} + X_{air} dh_{air} + dX_{air} h_{air} + X_{liquid} dh_{liquid} + dX_{liq} h_{liquid}$$
$$- dR\ T - R\ dT$$

(n) (o)

(b)～(m)の計算方法は h_pTX_der を参照

algorithm の抜粋1

```
  u_der := X_steam*Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(
        data=steam,
        T=T,
        refChoice=ReferenceEnthalpy.UserDefined,
        h_off=46479.819 + 2501014.5,
        dT=dT) + dX_steam*Modelica.Media.IdealGases.Common.Functions.h_Tlow(
        data=steam,
        T=T,
        refChoice=ReferenceEnthalpy.UserDefined,
        h_off=46479.819 + 2501014.5) + X_air*
    Modelica.Media.IdealGases.Common.Functions.h_Tlow_der(
        data=dryair,
        T=T,
        refChoice=ReferenceEnthalpy.UserDefined,
        h_off=25104.684,
        dT=dT) + dX_air*Modelica.Media.IdealGases.Common.Functions.h_Tlow(
        data=dryair,
        T=T,
        refChoice=ReferenceEnthalpy.UserDefined,
        h_off=25104.684) + X_liquid*enthalpyOfWater_der(T=T, dT=dT) +
    dX_liq*enthalpyOfWater(T) - dR_gas*T - R_gas*dT;
```

(b)(c)

(d)(e)

(f)(g)

(h)(i)

(j)(k) (l)(m) (n)(o)

(o) $$R_{gas} = steam\,.\,R\frac{X_{steam}}{1-X_{liquid}} + dryair\,.\,R\frac{X_{air}}{1-X_{liquid}}$$

(n) $$dR_{gas} = steam\,.\,R\left\{\frac{dX_{steam}}{1-X_{liquid}} + \frac{dX_{liq}\ X_{steam}}{(1-X_{liquid})^2}\right\} + dryair\,.\,R\left\{\frac{dX_{air}}{1-X_{liquid}} + \frac{dX_{liq}\ X_{air}}{(1-X_{liquid})^2}\right\}$$

$$= \frac{steam\,.\,R\left\{dX_{steam}(1-X_{liquid}) + dX_{liq}X_{steam}\right\} + dryair\,.\,R\left\{dX_{air}(1-X_{liquid}) + dX_{liq}X_{air}\right\}}{(1-X_{liquid})^2}$$

algorithm の抜粋2

```
algorithm
  p_steam_sat := saturationPressure(T);
  x_sat := p_steam_sat*k_mair/max(100*Modelica.Constants.eps, p -
    p_steam_sat);
X_sat := min(x_sat*(1 - X[Water]), 1.0);
X_liquid := Utilities.spliceFunction(
      X[Water] - X_sat,
      0.0,
      X[Water] - X_sat,
      1e-6);
X_steam := X[Water] - X_liquid;
X_air := 1 - X[Water];
R_gas := steam.R*X_steam/(1 - X_liquid) + dryair.R*X_air/(1 - X_liquid);

dX_air := -dX[Water];
dps := saturationPressure_der(Tsat=T, dTsat=dT);
dx_sat := k_mair*(dps*(p - p_steam_sat) - p_steam_sat*(dp - dps))/(p -
  p_steam_sat)/(p - p_steam_sat);
dX_liq := Utilities.spliceFunction_der(
      X[Water] - X_sat,
      0.0,
      X[Water] - X_sat,
      1e-6,
      (1 + x_sat)*dX[Water] - (1 - X[Water])*dx_sat,
      0.0,
      (1 + x_sat)*dX[Water] - (1 - X[Water])*dx_sat,
      0.0);
dX_steam := dX[Water] - dX_liq;
dR_gas := (steam.R*(dX_steam*(1 - X_liquid) + dX_liq*X_steam) + dryair.R*
  (dX_air*(1 - X_liquid) + dX_liq*X_air))/(1 - X_liquid)/(1 - X_liquid);
```

(j) X_{liquid}

(b) X_{steam}

(f) X_{air}

(o) R_{gas}

(h) dX_{air}

(l) dX_{liq}

(d) dX_{steam}

(n) dR_{gas}

(4) 比エントロピの微分 (derivative of specific entropy)

比エントロピ(specific entropy) s_pTX(p, T ,X)

$$s = s_{0\_Tlow}(dryair, T)(1 - X[Water]) + s_{0\_Tlow}(steam, T)X[Water] + s_p$$

$$s_p = -R_{mol}\left[\text{smoothMax}\left(\frac{X[Water]}{MMX[Water]}\log\left(\frac{\max\left(Y[Water], \varepsilon\right)\, p}{p_{ref}}\right),\, 0.0,\, 1\times10^{-9}\right)\right.$$

$$\left. + \text{smoothMax}\left(\frac{1 - X[Water]}{MMX[Air]}\log\left(\frac{\max\left(Y[Air], \varepsilon\right)\, p}{p_{ref}}\right),\, 0.0,\, 1\times10^{-9}\right)\right]$$

比エントロピの微分(derivative specific entropy)

$$\Rightarrow ds = \quad s_{0\_Tlow\_der}(dryair, T, dT)\ (1 - X[Water]) \quad \text{(a)}$$

$$+ s_{0\_Tlow\_der}(steam, T, dT)\ X[Water] \quad \text{(b)}$$

$$+ s_{0\_Tlow}(dryair, T)\ dX[Air] \quad \text{(c)}$$

$$+ s_{0\_Tlow}(steam, T)\ dX[Water] \quad \text{(d)}\ \text{(e)}\ \text{(f)}$$

$$-R_{mol}\left[\frac{\text{smoothMax\_der}(s_{water}, 0.0, 1e-9, ds_{water}, 0, 0)}{MMX[Water]}\right. \quad \text{(g)}\ \text{(h)}$$

$$\left. + \frac{\text{smoothMax\_der}(s_{air}, 0.0, 1e-9, ds_{air}, 0, 0)}{MMX[Air]}\right]$$

(a), (b) 比エントロピの温度依存項の微分

Modelica.Media.IdealGases.Common.Functions.s0_Tlow_der(data,T,T_der)

$$C_p(T) = R\sum_{i=1}^{7} a_i T^{i-3}$$

$$S_0(T) = R\left(-\frac{a_1}{2T^2} - \frac{a_2}{T} + a_3\log(T) + \sum_{i=4}^{7}\frac{T^{i-3}}{i-3} + b_2\right)$$

$$\left(\frac{\partial s}{\partial T}\right)_p = \frac{Cp}{T} = R\sum_{i=1}^{7} a_i T^{i-4} = R\left(a_1T^{-3} + a_2T^{-2} + a_3T^{-1} + \sum_{i=4}^{7} a_i T^{i-4}\right)$$

$$\Rightarrow S_0(T) = R\left(\int \frac{Cp}{T}dT + b_2\right) = R\left(-\frac{a_1}{2T^2} - \frac{a_2}{T} + a_3\log(T) + \sum_{i=4}^{7}\frac{T^{i-3}}{i-3} + b_2\right)$$

$$\therefore \text{s0\_Tlow\_der}(data, T, T\_der) = \left(\frac{\partial s}{\partial T}\right)_p T\_der = \frac{\text{cp\_Tlow}(data, T)}{T} T\_der$$

Enthalpy and entropy are obtained by integrating $C_p^o(T)$ and $C_p^o(T)/T$, respectively, with respect to T

NASA Glenn Coefficients for Calculating Thermodynamic Properties of Individual Species

https://www.grc.nasa.gov/WWW/CEAWeb/TP-2002-211556.pdf p.2

MSL 3.2.3 では、
s0_Tlow と同じ式に
なっている?

https://github.com/modelica/ModelicaStandardLibrary/issues/2870

(e) $$s_{water} = X[Water]\ \log\left(\frac{\max(Y[Water], \varepsilon)p}{p_{ref}}\right)$$

(f) $$ds_{water} = \left[\log\left(\frac{\max(Y[Water], \varepsilon)\ p}{p_{ref}}\right) + \left\{\frac{X[Water]}{Y[Water]}\left(\frac{MMX[Air]\ MMX[Water]}{(X[Air]\ MMX[Water] + X[Water]\ MMX[Air])^2}\right)\right\}\right] dX[Water] + X[Water]\frac{1}{p} dp$$

(g) $$s_{air} = (1 - X[Water])\ \log\left(\frac{\max(Y[Air], \varepsilon)\ p}{p_{ref}}\right)$$

(h) $$ds_{air} = \left[\log\left(\frac{\max(Y[Air], \varepsilon)\ p}{p_{ref}}\right) + \left\{\frac{X[Air]}{Y[Air]}\left(\frac{MMX[Water]\ MMX[Air]}{(X[Air]\ MMX[Water] + X[Water]\ MMX[Air])^2}\right)\right\}\right]\ dX[Air] + X[Air]\frac{1}{p}\ dp$$

Derivative of mol fractions

$$Y[Water] = \frac{\frac{X[Water]}{MMX[Water]}}{\frac{X[Water]}{MMX[Water]} + \frac{X[Air]}{MMX[Air]}} = \frac{X[Water]\ MMX[Air]}{X[Water]MMX[Air] + (1 - X[Water])MMX[Water]}$$

$$X[Air] = 1 - X[Water]$$

$$\frac{dY[Water]}{dX[Water]} = \frac{MMX[Air]}{X[Water]MMX[Air] + (1 - X[Water])MMX[Water]} - \frac{X[Water]\ MMX[Air]\ (MMX[Air] - MMX[Water])}{\{X[Water]MMX[Air] + (1 - X[Water])MMX[Water]\}^2}$$

$$= \frac{MMX[Air]\{X[Water]MMX[Air] + (1 - X[Water])MMX[Water]\} - \{X[Water]\ MMX[Air]\ (MMX[Air] - MMX[Water])\}}{\{X[Water]MMX[Air] + (1 - X[Water])MMX[Water]\}^2}$$

$$= \frac{MMX[Air]^2\ X[Water] + MMX[Air]MMX[Water] - MMX[Air]\ MMX[Water]\ X[Water] - MMX[Air]^2\ X[Water] + MMX[Air]\ MMX[Water]\ X[Water]}{\{X[Water]MMX[Air] + (1 - X[Water])MMX[Water]\}^2}$$

$$= \frac{MMX[Air]MMX[Water]}{\{X[Water]MMX[Air] + X[Air]MMX[Water]\}^2}$$

$$Y[Air] = \frac{\frac{X[Air]}{MMX[Air]}}{\frac{X[Water]}{MMX[Water]} + \frac{X[Air]}{MMX[Air]}} = \frac{X[Air]\ MMX[Water]}{(1 - X[Air])MMX[Air] + X[Air]MMX[Water]}$$

$$X[Water] = 1 - X[Air]$$

$$\frac{dY[Air]}{dX[Air]} = \frac{MMX[Water]}{(1 - X[Air])MMX[Air] + X[Air]MMX[Water]} - \frac{X[Air]\ MMX[Water](-MMX[Air] + MMX[Water])}{\{(1 - X[Air])MMX[Air] + X[Air]MMX[Water]\}^2}$$

$$= \frac{MMX[Water]\{(1 - X[Air])MMX[Air] + X[Air]MMX[Water]\} - \{X[Air]\ MMX[Water](-MMX[Air] + MMX[Water])\}}{\{(1 - X[Air])MMX[Air] + X[Air]MMX[Water]\}^2}$$

$$= \frac{MMX[Water]MMX[Air] - MMX[Water]MMX[Air]X[Air] + MMX[Water]^2X[Air] + MMX[Water]MMX[Air]X[Airr] - MMX[Water]^2\ X[Air]}{\{(1 - X[Air])MMX[Air] + X[Air]MMX[Water]\}^2}$$

$$= \frac{MMX[Water]MMX[Air]}{\{X[Water]\ MMX[Air] + X[Air]MMX[Water]\}^2}$$

Derivation of (f) from (e)

(e)

$$s_{water} = X[Water]\ \log\left(\frac{\max(Y[Water],\varepsilon)\ p}{p_{ref}}\right) = X[Water]\left(\log\left(\max(Y[Water],\varepsilon)\right) + \log\left(\frac{p}{p_{ref}}\right)\right)$$

$$= X[Water]\ \log\left(\max(Y[Water],\varepsilon)\right) + X[Water]\ \log\left(\frac{p}{p_{ref}}\right)$$

for $Y[Water] \geq \varepsilon$

$$d\left(\log(\max(Y[Water],\varepsilon))\right) \rightarrow d\left(\log(Y[Water])\right)$$

$$= \frac{1}{Y[Water]}\frac{dY[Water]}{dX[Water]}dX[Water]$$

(f)

$$ds_{water} = dX[Water]\ \log\left(\max(Y[Water],\varepsilon)\right) + X[Water]\ d\left(\log\left(\max(Y[Water],\varepsilon)\right)\right)$$

$$+ dX[Water]\ \log\left(\frac{p}{p_{ref}}\right) + X[Water]\ d\left(\log\left(\frac{p}{p_{ref}}\right)\right)$$

$$= \left(\log(\max(Y[Water],\varepsilon)) + \frac{X[Water]}{Y[Water]}\frac{dY[Water]}{dX[Water]} + \log\left(\frac{p}{p_{ref}}\right)\right)dX[Water] + X[Water]\frac{\frac{dp}{p_{ref}}}{\frac{p}{p_{ref}}}$$

$$= \left(\log\left(\max(Y[Water],\varepsilon)\frac{p}{p_{ref}}\right) + \frac{X[Water]}{Y[Water]}\frac{MMX[Air]\ MMX[Water]}{\{X[Water]MMX[Air] + X[Air]MMX[Water]\}^2}\right)dX[Water]$$

$$+ X[Water]\frac{1}{p}dp$$

Derivation of (h) from (g)

(g)

$$s_{air} = (1 - X[Water])\ \log\left(\frac{\max(Y[Air], \varepsilon)\ p}{p_{ref}}\right)$$

$$= X[Air]\ \log\left(\max(Y[Air], \varepsilon)\right) + X[Air]\ \log\left(\frac{p}{p_{ref}}\right)$$

$$d\left(\log\left(\max(Y[Air], \varepsilon)\right)\right) \to d\left(\log(Y[Air])\right) \quad \textbf{for}\ \ Y[Air] \geq \varepsilon$$

(h)

$$ds_{air} = dX[Air]\ \log\left(\max(Y[Air], \varepsilon)\right) + X[Air]\ d\left(\log\left(\max(Y[Air], \varepsilon)\right)\right) = \frac{1}{Y[Air]}\frac{dY[Air]}{dX[Air]}dX[Air]$$

$$+ dX[Air]\ \log\left(\frac{p}{p_{ref}}\right) + X[Air]\ d\left(\log\left(\frac{p}{p_{ref}}\right)\right)$$

$$= \left(\log\left(\max(Y[Air], \varepsilon)\frac{p}{p_{ref}}\right) + \frac{X[Air]}{Y[Air]}\frac{dY[Air]}{dX[Air]}\right)dX[Air] + X[Air]\frac{\frac{dp}{p_{ref}}}{\frac{p}{p_{ref}}}$$

$$= \left(\log\left(\max(Y[Air], \varepsilon)\frac{p}{p_{ref}}\right) + \frac{X[Air]}{Y[Air]}\frac{MMX[Water]\,MMX[Air]}{\{X[Water]\ MMX[Air] + X[Air]MMX[Water]\}^2}\right)dX[Air]$$

$$+ X[Air]\frac{1}{p}dp$$

MoistAir.s_pTX_der(p,T,X,dp,dT,dX) in MSL3.2.3

```
protected
  MoleFraction[2] Y=massToMoleFractions(X, {steam.MM,dryair.MM})
    "Molar fraction";

algorithm
  ds := Modelica.Media.IdealGases.Common.Functions.s0_Tlow_der(
        dryair,
        T,
        dT)*(1 - X[Water]) +
    Modelica.Media.IdealGases.Common.Functions.s0_Tlow_der(
        steam,
        T,
        dT)*X[Water] + Modelica.Media.IdealGases.Common.Functions.s0_Tlow(
    dryair, T)*dX[Air] + Modelica.Media.IdealGases.Common.Functions.s0_Tlow(
    steam, T)*dX[Water] - Modelica.Constants.R*(1/MMX[Water]*
    Utilities.smoothMax_der(
        X[Water]*Modelica.Math.log(max(Y[Water], Modelica.Constants.eps)*p/
    reference_p),
      0.0,
      1e-9,
      (Modelica.Math.log(max(Y[Water], Modelica.Constants.eps)*p/
    reference_p) + (X[Water]/Y[Water]*(X[Air]*MMX[Water]/(X[Air]*MMX[
    Water] + X[Water]*MMX[Air])^2)))*dX[Water] + X[Water]*reference_p/p*
    dp,
      0,
      0) - 1/MMX[Air]*Utilities.smoothMax_der(
      (1 - X[Water])*Modelica.Math.log(max(Y[Air], Modelica.Constants.eps)
    *p/reference_p),
      0.0,
      1e-9,
      (Modelica.Math.log(max(Y[Air], Modelica.Constants.eps)*p/
    reference_p) + (X[Air]/Y[Air]*(X[Water]*MMX[Air]/(X[Air]*MMX[Water]
     + X[Water]*MMX[Air])^2)))*dX[Air] + X[Air]*reference_p/p*dp,
      0,
      0));
annotation ( …);
```

(a) (b) (c) (d) (e) (f) (g) (h)

bug? https://github.com/modelica/ModelicaStandardLibrary/issues/2874

```
protected
  MoleFraction[2] Y=massToMoleFractions(X, {steam.MM,dryair.MM})
    "Molar fraction";

algorithm
  ds := Modelica.Media.IdealGases.Common.Functions.s0_Tlow_der(
        dryair,
        T,
        dT)*(1 - X[Water]) +
    Modelica.Media.IdealGases.Common.Functions.s0_Tlow_der(
        steam,
        T,
        dT)*X[Water] + Modelica.Media.IdealGases.Common.Functions.s0_Tlow(
    dryair, T)*dX[Air] + Modelica.Media.IdealGases.Common.Functions.s0_Tlow(
    steam, T)*dX[Water] - Modelica.Constants.R*(1/MMX[Water]*
    Utilities.smoothMax_der(
        X[Water]*Modelica.Math.log(max(Y[Water], Modelica.Constants.eps)*p/
      reference_p),
        0.0,
        1e-9,
        (Modelica.Math.log(max(Y[Water], Modelica.Constants.eps)*p/
      reference_p) + (X[Water]/Y[Water]*(MMX[Air]*MMX[Water]/(X[Air]*MMX[
      Water] + X[Water]*MMX[Air])^2)))*dX[Water] + X[Water]/p*dp,
        0,
        0) + 1/MMX[Air]*Utilities.smoothMax_der(
        (1 - X[Water])*Modelica.Math.log(max(Y[Air], Modelica.Constants.eps)
      *p/reference_p),
        0.0,
        1e-9,
        (Modelica.Math.log(max(Y[Air], Modelica.Constants.eps)*p/
      reference_p) + (X[Air]/Y[Air]*(MMX[Water]*MMX[Air]/(X[Air]*MMX[Water]
       + X[Water]*MMX[Air])^2)))*dX[Air] + X[Air]/p*dp,
        0,
        0));
annotation ( …);
```

(a) (b) (c) (d) (e) (f) (g) (h)

− ⇒ +

(5) 定積比熱 (specific heat Cv)

$\varphi < 1$ で成立する

MoistAir.specificHeatCapacityCv(state) の計算式

$$
\begin{aligned}
c_v = \quad & \mathrm{cp\_Tlow}(dryair, state.T)\ (1 - state.X[Water]) \\
& + \mathrm{cp\_Tlow}(steam, state.T)\ state.X[Water] \\
& + \mathrm{gasConstant}(state)
\end{aligned}
$$

```
  cv := Modelica.Media.IdealGases.Common.Functions.cp_Tlow(dryair, state.T)
    *(1 - state.X[Water]) +
    Modelica.Media.IdealGases.Common.Functions.cp_Tlow(steam, state.T)*
    state.X[Water] - gasConstant(state);
```

(6) 等エントロピー指数 (isentropic exponent)

MoistAir.isentropicExponent(state) の計算式

$\varphi < 1$ で成立する

$$
\gamma = \frac{\mathrm{specificHeatCapacity\_Cp}(state)}{\mathrm{specificHeatCapacity\_Cv}(state)}
$$

```
  algorithm
    gamma := specificHeatCapacityCp(state)/specificHeatCapacityCv(state);
```

(7) 等圧熱膨張係数 (isobaric expansion coefficient)

MoistAir.isobaricExpansionCoefficient(state)

$$\beta = \frac{1}{\text{temperature}(state)}$$

```
algorithm
  beta := 1/temperature(state);
```

(8) 等温圧縮率 (isothermal compressibility)

MoistAir.isothermalCompressiblility(state)

$$\kappa = \frac{1}{\text{pressure}(state)}$$

```
algorithm
  kappa := 1/pressure(state);
```

(9) 音速 (velocity of sound)

$\varphi < 1$ で成立する

MoistAir.velocityOfSound(state)

$$a = \sqrt{\text{isentropicExponent}(state)\ \text{gasConstant}(state)\ \text{temperature}(state)}$$

```
algorithm
  a := sqrt(isentropicExponent(state)*gasConstant(state)*temperature(state));
```

(10) ギブスエネルギー (specific Gibbs energy)

MoistAir.specificGibbsEnergy(state)

$g = h - Ts$ $\varphi < 1$ で成立する

$$g = \text{h\_pTX}(state.p, state.T, state.X) - state.T \ \text{specificEntropy}(state)$$

algorithm の抜粋

```
  algorithm
    g := h_pTX(
          state.p,
          state.T,
          state.X) - state.T*specificEntropy(state);
    annotation (smoothOrder=2, Documentation(info="<html> ...);
  end specificGibbsEnergy;
```

(11) ヘルムホルツエネルギー (specific Helmholtz energy)

MoistAir.specificHelmholtzEnergy(state)

$f = h - RT - Ts$ $\varphi < 1$ で成立する

$$f = \text{h\_pTX}(p, T, X) - gasConstant(state)\ state.T - state.T \ \text{specificEntropy}(state)$$

algorithm の抜粋

```
  algorithm
    f := h_pTX(
          state.p,
          state.T,
          state.X) - gasConstant(state)*state.T - state.T*specificEntropy(
      state);
    annotation (smoothOrder=2, Documentation(info="<html> ...);
  end specificHelmholtzEnergy;
```

(12) 等エントロピ過程のエンタルピ

状態 refState から圧力 p_downsteam に等エントロピ的に変化したときの比エンタルピ

① MoistAir.isentripicEnthalpy(p_downstream, refState)

$$h_{is} = \mathrm{h\_pTX}\left(p_{downstream}, \mathrm{T\_psX}\left(p_{downstream}, \mathrm{specificEntropy}(refState), refstate.X\right), refstate.X\right)$$

状態 refState の比エントロピ

状態 refState から圧力 p_downstream に等エントロピ的に変化したときの温度

algorithm の抜粋

```
algorithm
  h_is := Modelica.Media.Air.MoistAir.h_pTX(
        p_downstream,
        Modelica.Media.Air.MoistAir.T_psX(
          p_downstream,
          Modelica.Media.Air.MoistAir.specificEntropy(refState),
          refState.X),
        refState.X);
```

②MoistAir.isentripicEnthalpyApproximation(ps, state)

$$h_{is} = h + \frac{\gamma}{\gamma - 1} RT \left(\left(\frac{p_2}{p} \right)^{(\gamma-1)/\gamma} - 1 \right)$$

algorithm の抜粋

```
protected
  SpecificEnthalpy h "Specific enthalpy at upstream location";
  IsentropicExponent gamma=isentropicExponent(state) "Isentropic exponent";
algorithm
  X := state.X;
//  X := if reducedX then cat(1,state.X,{1-sum(state.X)}) else state.X;
h := {Modelica.Media.IdealGases.Common.Functions.h_Tlow(
      data=steam,
      T=state.T,
      refChoice=ReferenceEnthalpy.UserDefined,
      h_off=46479.819 + 2501014.5),
  Modelica.Media.IdealGases.Common.Functions.h_Tlow(
      data=dryair,
      T=state.T,
      refChoice=ReferenceEnthalpy.UserDefined,
      h_off=25104.684)}*X;

h_is := h + gamma/(gamma - 1.0)*(state.T*gasConstant(state))*((p2/state.p)
  ^((gamma - 1)/gamma) - 1.0);
```

γ

水蒸気の比エンタルピ

乾燥空気の比エンタルピ

(13) 密度の偏微分

$$\rho(p,T)=\frac{p}{RT} \qquad \Rightarrow d\rho=\frac{1}{RT}dp-\frac{p}{RT^2}dT=\frac{1}{RT}dp-\frac{\rho}{T}dT$$

$$\left(\frac{\partial\rho}{\partial p}\right)_T=\frac{1}{RT}, \qquad \left(\frac{\partial\rho}{\partial T}\right)_p=-\frac{\rho}{T}$$

MoistAir.density_derp_T(state) の algorithm の抜粋

```
ddpT := 1/(gasConstant(state)*temperature(state));
```

MoistAir.density_derT_p(state) の algorithm の抜粋

```
ddTp := -density(state)/temperature(state);
```

$$C_p = \left(\frac{\partial h}{\partial T}\right)_p \Rightarrow dT = \frac{1}{c_p}dh$$

$$d\rho = \frac{1}{RT}dp - \frac{p}{RT^2}dT = \frac{1}{RT}dp - \frac{\rho}{T}dT = \frac{1}{RT}dp - \frac{\rho}{C_pT}dh$$

$$\left(\frac{\partial \rho}{\partial p}\right)_h = \frac{1}{RT}, \quad \left(\frac{\partial \rho}{\partial h}\right)_p = -\frac{\rho}{C_pT}$$

MoistAir.density_derp_h(state) の algorithm の抜粋

```
    ddph := 1/(gasConstant(state)*temperature(state));
```

MoistAir.density_derh_p(state) の algotithm の抜粋

```
    ddhp := -density(state)/(specificHeatCapacityCp(state)*temperature(state));
```

MoistAir.density_derX(state)

$$\rho = \frac{p}{RT} \Rightarrow \frac{\partial \rho}{\partial X_i} = -\frac{p}{R^2 T}\frac{\partial R}{\partial X_i} = \frac{-p\frac{\partial R}{\partial X_i}}{R^2 T}$$

$$\begin{aligned}
R &= steam.R \cdot X[Water] + dryair.R \cdot (1 - X[Water]) \\
&= (steam.R - dryair.R) \cdot X[Water] + dryair.R \\
-p\frac{dR}{dX[Water]} &= -\,\mathrm{pressure}(state)(steam.R - dryair.R) \\
R^2T &= \Big((steam.R - dryair.R)X[Water] + dryair.R\Big)^2 \,\mathrm{temperature}(state) \\
\mathrm{dddX}[Water] &= \frac{-p\frac{\partial R}{\partial X[Water]}}{R^2T} = \frac{-\mathrm{pressure}(state)(steam.R - dryair.R)}{\Big((steam.R - dryair.R)X[Water] + dryair.R\Big)^2 \,\mathrm{temperature}(state)}
\end{aligned}$$

$$\begin{aligned}
R &= steam.R(1 - X[Air]) + dryair.R \cdot X[Air] \\
&= steam.R + (dryair.R - steam.R)X[Air] \\
-p\frac{dR}{dX[Air]} &= -\,\mathrm{pressure}(state)(dryair.R - steam.R) \\
R^2T &= (steam.R + (dryair.R - steam.R) \cdot X[Air])^2 \mathrm{temperature(state)} \\
\mathrm{dddX}[Air] &= \frac{-p\frac{\partial R}{\partial X[Air]}}{R^2T} = \frac{-\mathrm{pressure}(state)(dryair.R - steam.R)}{(steam.R + (dryair.R - steam.R) \cdot X[Air])^2 \mathrm{temperature(state)}}
\end{aligned}$$

MoistAir.density_derX(state) の algorithm (MSL 3.2.3)

```
  algorithm
    dddX[Water] := pressure(state)*(steam.R - dryair.R)/((steam.R - dryair.R)
      *state.X[Water]*temperature(state) + dryair.R*temperature(state))^2;
    dddX[Air] := pressure(state)*(dryair.R - steam.R)/((dryair.R - steam.R)*
      state.X[Air]*temperature(state) + steam.R*temperature(state))^2;
```

X [kg/kg] は無次元だから、$\frac{\partial \rho}{\partial X_i}$ は ρ と同じ次元となる。

上式の次元は $\frac{pR}{(RT)^2} = \frac{p}{RT^2} = \frac{\rho}{T}$ となってしまう。 bug?

https://github.com/modelica/ModelicaStandardLibrary/issues/2868

```
  algorithm
    dddX[Water] := - pressure(state)*(steam.R - dryair.R)/((steam.R - dryair.R)
      *state.X[Water] + dryair.R)^2/temperature(state)
    dddX[Air] := -pressure(state)*(dryair.R - steam.R)/((dryair.R - steam.R)*
      state.X[Air] + steam.R)^2/temperature(state);
```

(14) 粘性率 (viscosity)

粘性率と熱伝導率は DryAirNasa と同じ。

MoistAir.dynamicViscosity(state)

```
algorithm
  eta := 1e-6*Polynomials_Temp.evaluateWithRange(
      {9.7391102886305869E-15,-3.1353724870333906E-11,4.3004876595642225E-08,
      -3.8228016291758240E-05,5.0427874367180762E-02,1.7239260139242528E+01},
      Cv.to_degC(123.15),
      Cv.to_degC(1273.15),
      Cv.to_degC(state.T))
```

$$\eta = \sum_{j=1}^{n} p_j T^{n-j},\ T_{min} \le T \le T_{max}$$

(15) 熱伝導率 (thermal conductivity)

MoistAir.thermalConductivity(state)

```
algorithm
  lambda := 1e-3*Polynomials_Temp.evaluateWithRange(
      {6.5691470817717812E-15,-3.4025961923050509E-11,5.3279284846303157E-08,
      -4.5340839289219472E-05,7.6129675309037664E-02,2.4169481088097051E+01},
      Cv.to_degC(123.15),
      Cv.to_degC(1273.15),
      Cv.to_degC(state.T));
```

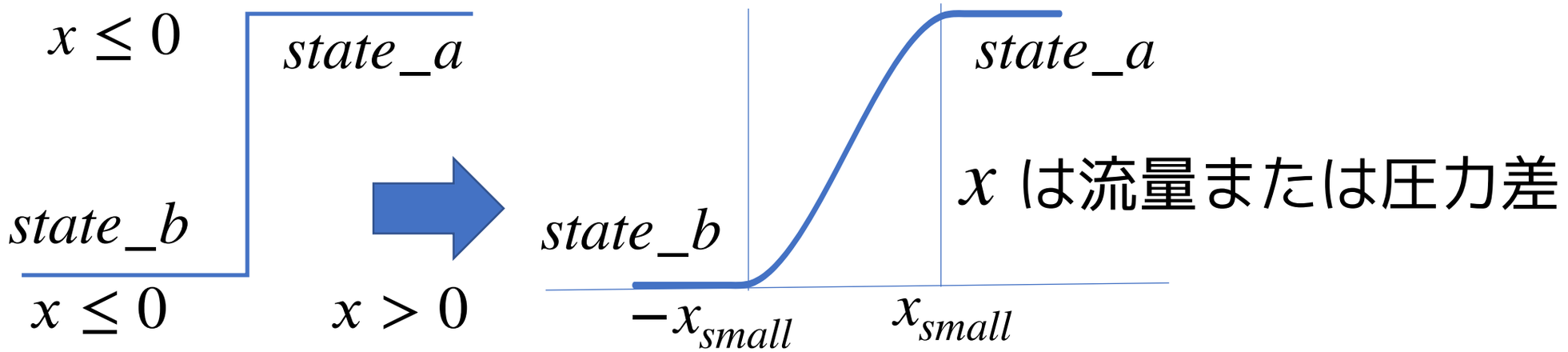
$$\lambda = \sum_{j=1}^{n} p_j T^{n-j},\ T_{min} \le T \le T_{max}$$

(16) 熱力学的状態の補完関数 (smooth state function)

x=0 で不連続な熱力学的状態 state_a, state_b を滑らかに補間する。

$$y = \begin{cases} state\_a, & x > 0 \\ state\_b, & x \le 0 \end{cases}$$

$$y = \mathrm{setSmoothState}(x, state\_a, state\_b, x_{small})$$



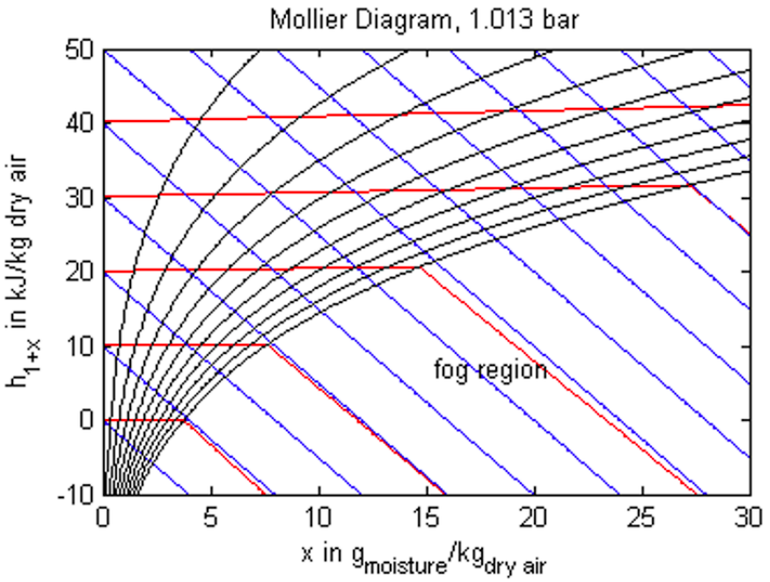
MoistAir.setSmoothState(x, state_a, state_b, x_small)

```
algorithm
  state := ThermodynamicState(
        p=Media.Common.smoothStep(
          x,
          state_a.p,
          state_b.p,
          x_small),
        T=Media.Common.smoothStep(
          x,
          state_a.T,
          state_b.T,
          x_small),
        X=Media.Common.smoothStep(
          x,
          state_a.X,
          state_b.X,
          x_small));
end setSmoothState;
```

EX.5 湿り空気線図のデータをつくる　PsychrometricData

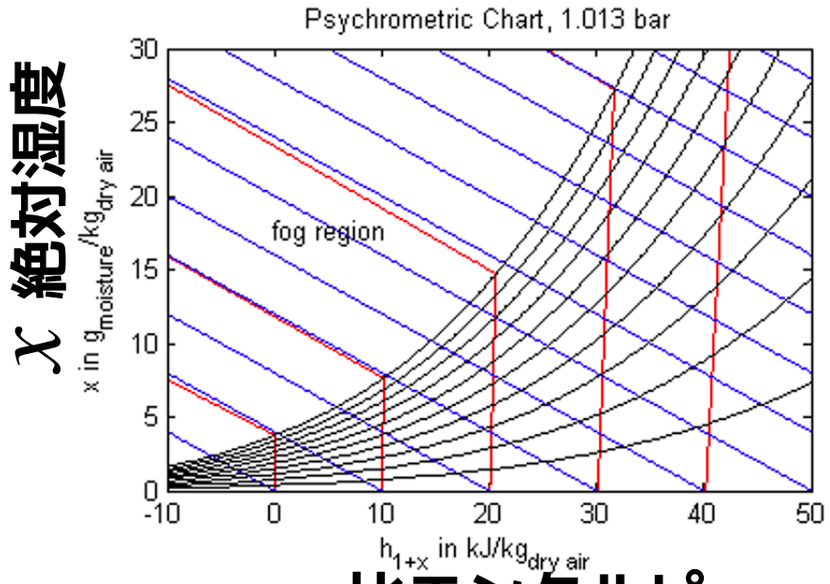
Modelica.Media.Examples.PsychrometricData のドキュメントより

(1) 湿り空気のチャートは、主に２種類ある。



比エンタルピ　h

x　絶対湿度



x 絶対湿度

比エンタルピ　h

互いに縦軸と
横軸を交換したもの

このモデルでは

x が時間的に変化する

$$t : 0 \rightarrow 1$$

$$x : x_{min} \rightarrow x_{max}$$

➡ **モリエ線図的な**
チャートが描ける

モリエ線図 (Mollier Diagram)　　湿り空気線図 (Psychrometric Chart)

ヨーロッパ各国で広く使用される。アメリカで一般的に使用される。

(2) チャートのデータ

チャートにプロットするデータは軸が傾いている!!

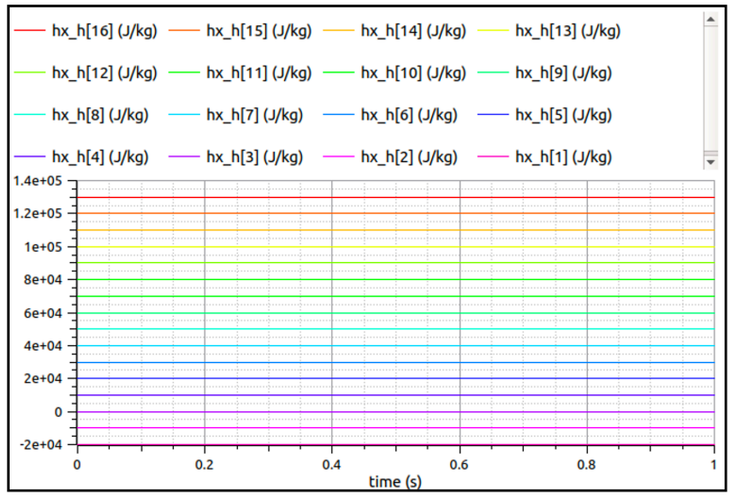
青：比エンタルピー定　hx_h[i]

赤：温度一定　hx_T[i]

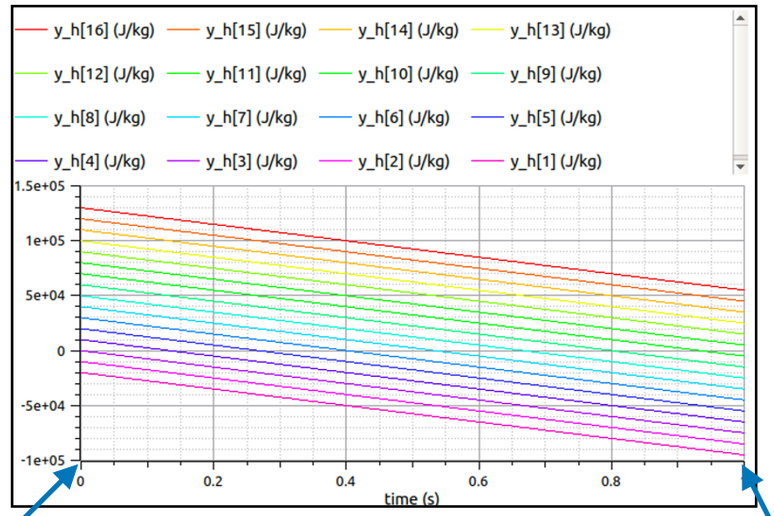
黒：相対湿度一定　hx_phi[i]

```
y_h[i] = hx_h[i] - diagSlope*x
y_T[i] = hx_T[i] - diagSlope*x
y_phi[i] = hx_phi[i] - diagSlope*x
```

diagSlope = enthalpyOfVaporization(273.15)　(0℃の気化熱)



等エンタルピ線　hx_h[i]



チャートの表示　y_h[i]

x_min = 0

x_max = 0.03

(3) パラメータ (parameters)

```
p_const =1e5   圧力
time=1         シミュレーション時間
x_min=0.00     t=0 の絶対湿度
x_max=0.03     t=1 の絶対湿度

n_T=11         等温度の数
T_min=253.15   温度の最小値
T_step=10      温度の間隔

n_h=16         等エンタルピ線の数
h_min=-20e3    比エンタルピの最小値
h_step=1e4     比エンタルピの間隔

n_phi=10       等相対湿度線の数
phi_min=0.1    相対湿度の最小値
h_step=0.1     相対湿度の間隔
```

等値線の値

温度

Variables	Value
☑ T_min	-20
▾ T_const	
[11]	80
[10]	70
[9]	60
[8]	50
[7]	40
[6]	30
[5]	20
[4]	10
[3]	0
[2]	-10
[1]	-20

比エンタルピ

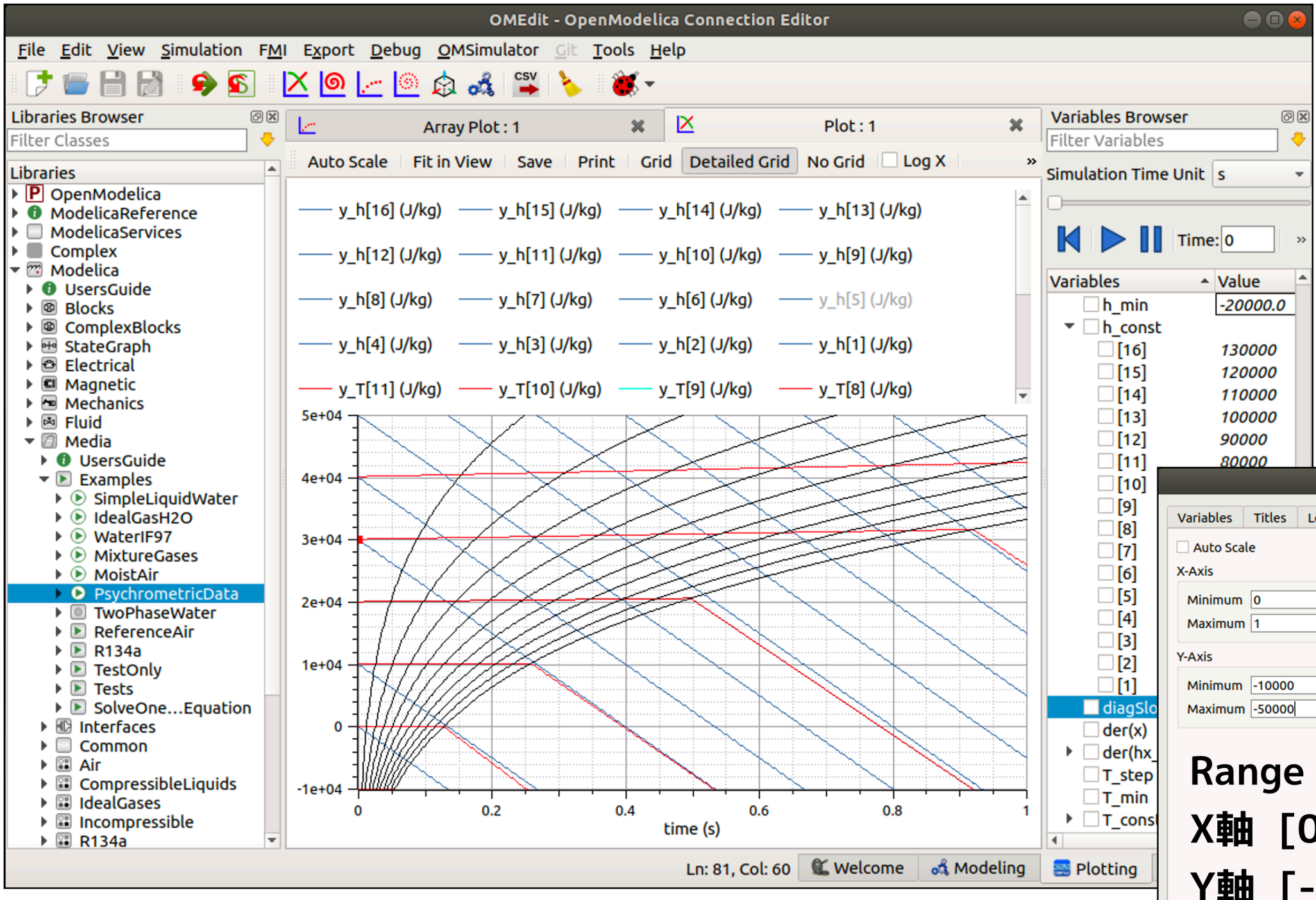
Variables	Value
▾ h_const	
[16]	130000
[15]	120000
[14]	110000
[13]	100000
[12]	90000
[11]	80000
[10]	70000
[9]	60000
[8]	50000
[7]	40000
[6]	30000
[5]	20000
[4]	10000
[3]	0
[2]	-10000
[1]	-20000

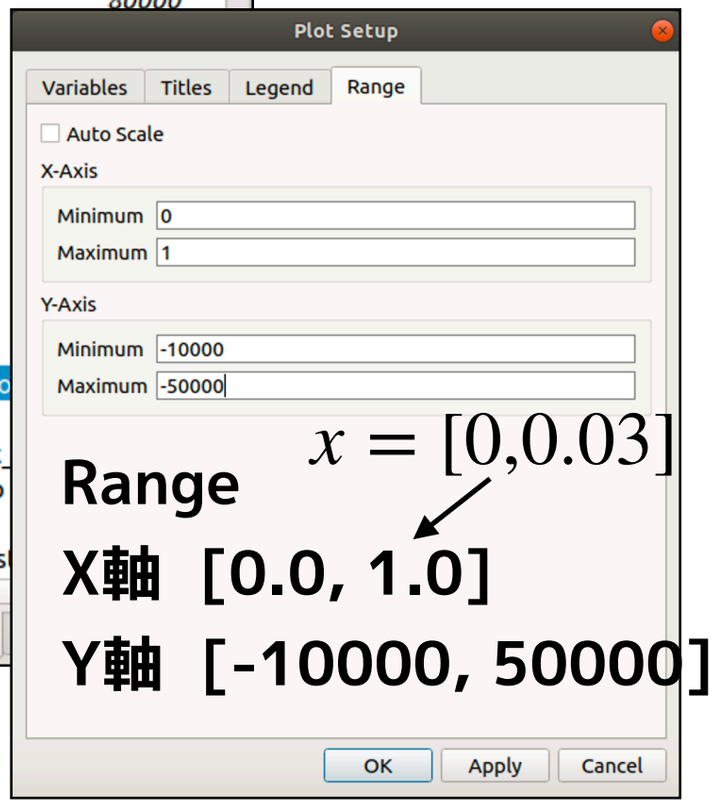
相対湿度

Variables	Value
▾ phi_const	
[10]	1
[9]	0.9
[8]	0.8
[7]	0.7
[6]	0.6
[5]	0.5
[4]	0.4
[3]	0.3
[2]	0.2
[1]	0.1

OpenModelica (OMEdit) を使ったチャートの作図

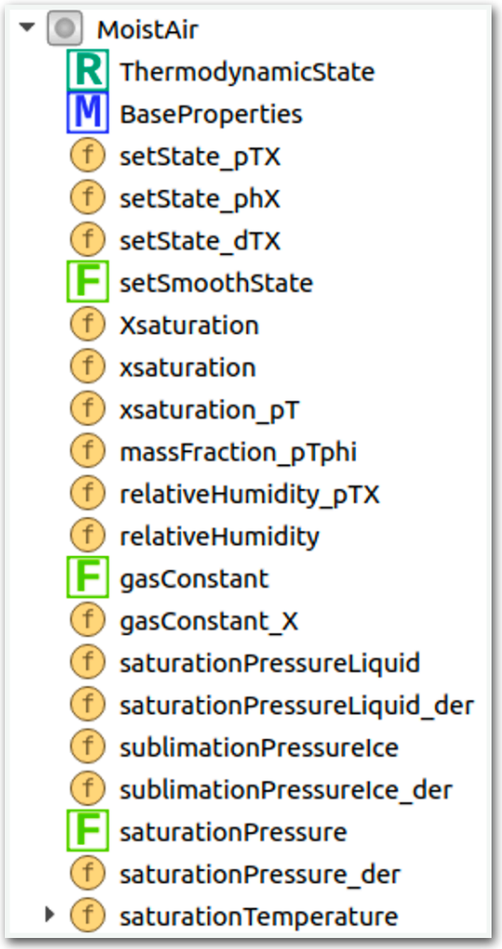
y_h[i]
y_T[i]
y_phi[i]
をプロットする



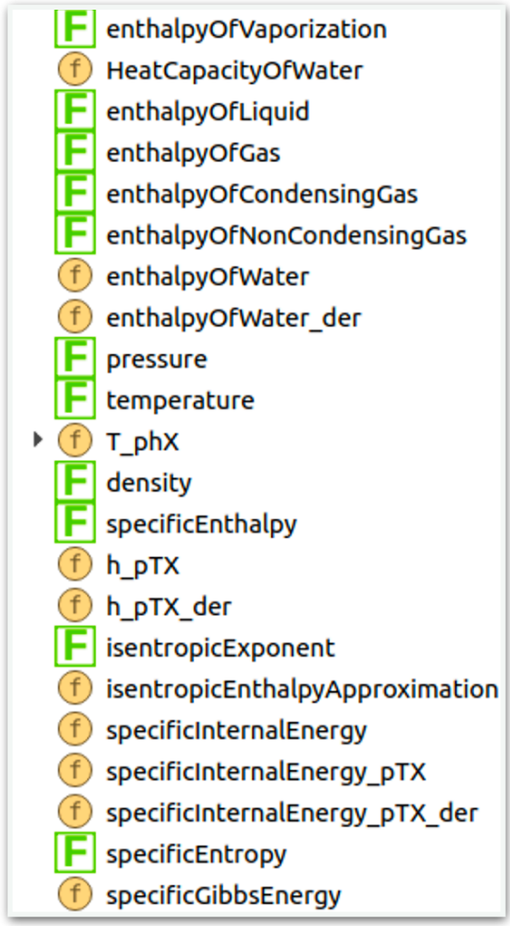


等値線の色は適宜変更した。

Modelica.Media.Air.MoistAir の クラス一覧(1)

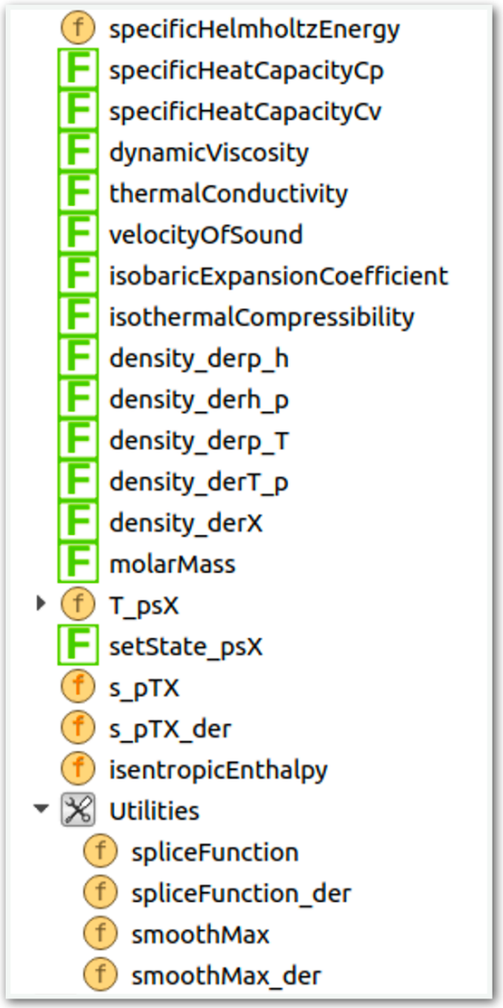


- ThermodynamicState
- BaseProperties
- setState_pTX
- setState_phX
- setState_dTX
- setSmoothState
- Xsaturation
- xsaturation
- xsaturation_pT
- massFraction_pTphi
- relativeHumidity_pTX
- relativeHumidity
- gasConstant
- gasConstant_X
- saturationPressureLiquid
- saturetionPressureLiquid_der
- sublimationPressureIce
- sublimationPressureIce_der
- saturationPressure
- saturationPressure_der
- saturationTemperature



- enthalpyOfVaporization
- HeatCapacityOfWater
- enthalpyOfLiquid
- enthalpyOfCondensingGas
- enthalpyOfNonCondensingGas
- enthalpyOfWater
- enthalpyOfWater_der
- pressure
- temperature
- T_phX
- density
- specificEnthalpy
- h_pTX
- h_pTX_der
- isentropicExponent
- isentropicEnthalpyApproximation
- specificInternalEnergy
- specificInternalEnergy_pTX
- specificInternalEnergy_pTX_der
- specificEntropy
- specificGibbsEnergy

Modelica.Media.Air.MoistAir の クラス一覧(2)



- specificHelmholtzEnergy
- specificHeatCapacityCp
- specificHeatCapacityCv
- dynamicViscosity
- thermalConductivity
- velocityOfSound
- isobaricExpansionCoefficient
- isothermalCompressibility
- density_derp_h
- density_derh_p
- density_derp_T
- density_derT_p
- density_derX
- molarMass
- T_psX
- setState_psX
- s_pTX
- s_pTX_der
- isentropicEnthalpy
- spliceFunction
- spliceFunction_der
- smoothMax
- smoothMax_der

まとめ

MoistAirは、乾燥空気 (dry air) と水 (water) から構成される湿り空気のモデルである。水分がさらに水蒸気 (steam) と凝縮水 (liquid water または ice water)に分かれ、凝縮水が乾燥空気と水蒸気の中に分散した霧の領域 (fog region) もモデル化可能である。以下のような特徴を持つ。

- 水蒸気と凝縮水の割合は、水の質量分率、温度、圧力に依存して変化し、水の分圧と飽和蒸気圧の関係から計算される。
- 気体部分（乾燥空気と水蒸気）は、SingleGasNasa の DryAirNasa と H2O の混合物である。
- 凝縮水部分は、体積を無視し、質量や比熱、比エンタルピ、比エントロピなどを考慮する。
- 粘性率と熱伝導率は、DryAirNasa と同じ値を用いる。
- 動的解析に必要な、飽和蒸気圧曲線の微分関数や熱力学的状態変数の微分関数も多数含まれる。

Modelica Standard Library License

https://github.com/modelica/ModelicaStandardLibrary/blob/master/LICENSE