OpenModelica講習中級
Modelica.Fluidライブラリ解説

付録1. パイプの圧力損失と質量流量
付録2. PartialVessel の圧力計算式

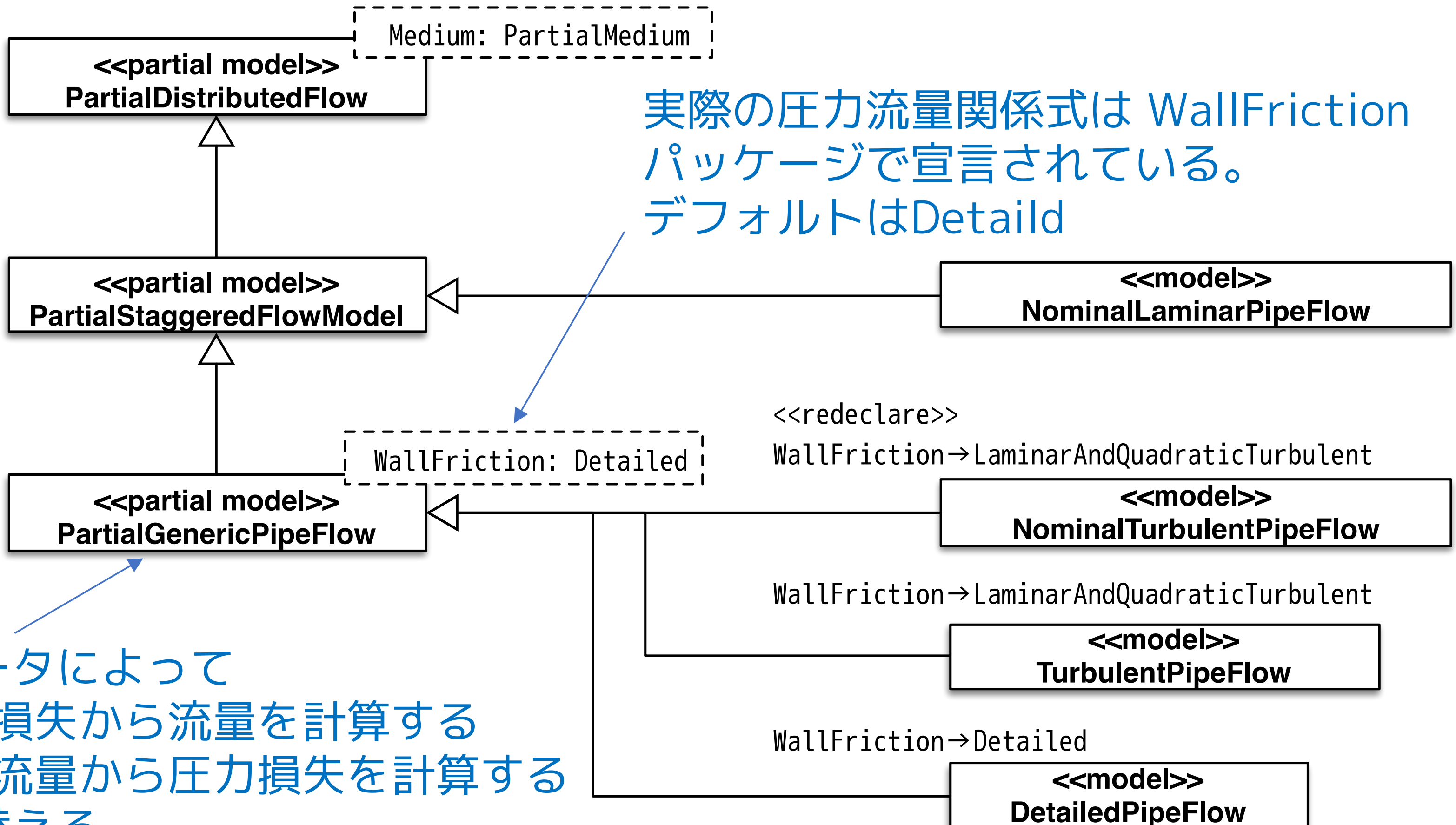
2017年12月7日 田中 周(有限会社アマネ流研)

付録1. パイプの圧力損失と質量流量

(1)概要
(2)計算方法を場合分けするパラメータ
(3)ホモトピー法とホモトピーオペレータ
(4) WallFriction.Detailed
- massFlowRate_dp
- pressureLoss_m_flow

(1) 概要

パイプの圧力損失と質量流量の関係は、PartialGenericFlow を継承したモデルで計算される。

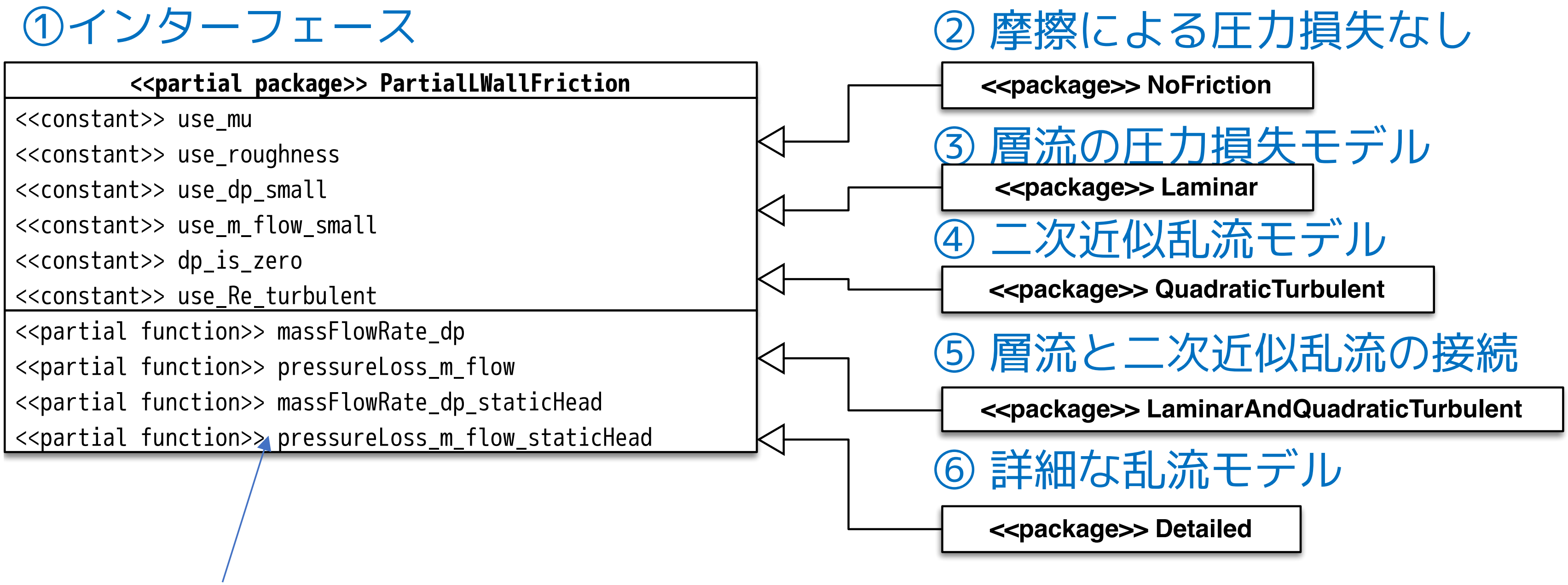


実際の圧力流量関係式は WallFriction パッケージで宣言されている。
デフォルトはDetaild

パラメータによって
- 圧力損失から流量を計算する
- 質量流量から圧力損失を計算する

を切り替える。

WallFriction パッケージ



順流から逆流に変わるとき圧力が連続的に変化する場合

- massFlowRate_dp: 圧力損失から質量流量を計算する
- pressureLoss_m_flow: 質量流量から圧力損失を計算する

不連続的に変化する場合やパイプ出入口に高低差がある場合

- massFlowRate_dp_staticHead: 圧力損失から質量流量を計算する
- pressureLoss_m_flow_staticHead: 質量流量から圧力損失を計算する

PartialGenericPipeFlow の質量流量または圧力損失の計算の部分

```
if continuousFlowReversal then
   // simple regularization
   if from_dp and not WallFriction.dp_is_zero then
     m_flows = homotopy(
       actual=  WallFriction.massFlowRate_dp(
                  dps_fg - {g*dheights[i]*rhos_act[i] for i in 1:n-1},
                  rhos_act,
                  rhos_act,
                  mus_act,
                  mus_act,
                  pathLengths_internal,
                  diameters,
                  (crossAreas[1:n-1]+crossAreas[2:n])/2,
                  (roughnesses[1:n-1]+roughnesses[2:n])/2,
                  dp_small/(n-1),
                  Res_turbulent_internal)*nParallel,
       simplified=  m_flow_nominal/dp_nominal*(dps_fg - g*dheights*rho_nominal));
   else
     dps_fg = homotopy(
       actual=  WallFriction.pressureLoss_m_flow(
                  m_flows/nParallel,
                  rhos_act,
                  rhos_act,
                  mus_act,
                  mus_act,
                  pathLengths_internal,
                  diameters,
                  (crossAreas[1:n-1]+crossAreas[2:n])/2,
                  (roughnesses[1:n-1]+roughnesses[2:n])/2,
                  m_flow_small/nParallel,
                  Res_turbulent_internal) + {g*dheights[i]*rhos_act[i] for i in 1:n-1},
       simplified=  dp_nominal/m_flow_nominal*m_flows + g*dheights*rho_nominal);
   end if;
```

(2) パラメータによる計算方法の場合分け

(3) ホモトピー法

```
 else
   // regularization for discontinuous flow reversal and static head
   if from_dp and not WallFriction.dp_is_zero then
     m_flows = homotopy(
       actual=  WallFriction.massFlowRate_dp_staticHead(
                  dps_fg,
                  rhos[1:n-1],
                  rhos[2:n],
                  mus[1:n-1],
                  mus[2:n],
                  pathLengths_internal,
                  diameters,
                  g*dheights,
                  (crossAreas[1:n-1]+crossAreas[2:n])/2,
                  (roughnesses[1:n-1]+roughnesses[2:n])/2,
                  dp_small/(n-1),
                  Res_turbulent_internal)*nParallel,
       simplified=  m_flow_nominal/dp_nominal*(dps_fg - g*dheights*rho_nominal));
   else
     dps_fg = homotopy(
       actual=  WallFriction.pressureLoss_m_flow_staticHead(
                  m_flows/nParallel,
                  rhos[1:n-1],
                  rhos[2:n],
                  mus[1:n-1],
                  mus[2:n],
                  pathLengths_internal,
                  diameters,
                  g*dheights,
                  (crossAreas[1:n-1]+crossAreas[2:n])/2,
                  (roughnesses[1:n-1]+roughnesses[2:n])/2,
                  m_flow_small/nParallel,
                  Res_turbulent_internal),
       simplified=  dp_nominal/m_flow_nominal*m_flows + g*dheights*rho_nominal);
   end if;
  end if;
```

（2）計算方法の場合分けのパラメータ

① from_dp
true なら圧力差から質量流量を計算する
flase なら質量流量から圧力差を計算する

momentumDynamics によって切り替わる

PartialGenericPipeFlow の宣言部

```
parameter Boolean from_dp = momentumDynamics >= Types.Dynamics.SteadyStateInitial
"= true, use m_flow = f(dp), otherwise dp = f(m_flow)”  annotation( ...);
```

momentumDynamics	from_dp	initial condition
DynamicFreeInitial	false	
FixedInitial	false	m_flow_start
SteadyStateInitial	true	d(m_flow)/dt = 0
SteadyState	true	

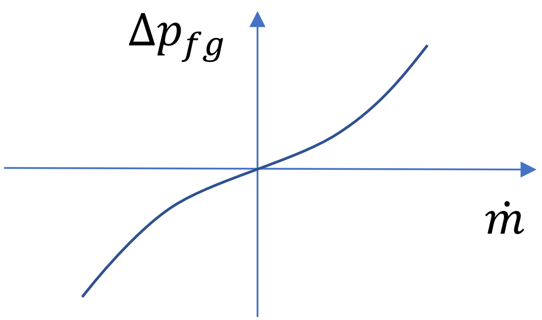
② WallFriction.dp_is_zero
粘性圧力損失が存在しない場合 true

③ continuousFlowReversal
順流から逆流に変化するとき圧力が連続的に変化する場合 true

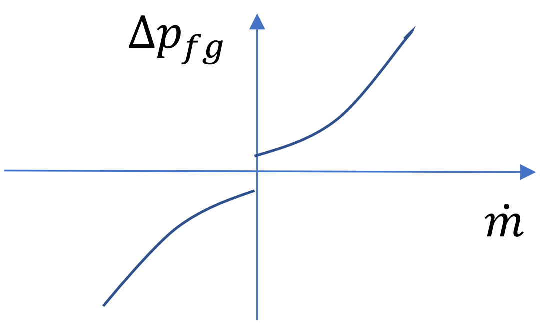
PartialGenericPipeFlow の宣言部

```
final parameter Boolean continuousFlowReversal=
    (not useUpstreamScheme)
    or constantPressureLossCoefficient
    or not allowFlowReversal
```

true



false



より、
- 風上差分スキームで無い
- 圧力損失係数が一定
- 逆流を考慮しない

のいずれかの場合 true となる。

false の場合、特別な regularizationが
必要となるので場合分けする。

(3) ホモトピー法とホモトピーオペレータ

概要

- 動的シミュレーションの初期化フェーズでは、大きな非線形方程式系を繰り返し計算で解く必要がある場合がある。
- ホモトピー法は、まず、簡単に収束するモデル (simplified model) を解き、このモデルから実際のモデル(actual model) までモデルを連続的に変形させて解を追跡することによって非線形方程式系を解く方法である。

ホモトピー変換 (homotopy transformation)

次のような式で λ を 0 から 1 まで連続的に変化させる。

$$\lambda * actual + (1-\lambda) * simplified$$

Modelica のホモトピーオペレータ

ホモトピーオペレータは、非線形方程式に入力する変数に対して使用する。

ホモトピーオペレータ

$$m\_flows = homotpy(actual, simplified)$$

(4) WallFriction.Detailed

ダルシー・ワイズバッハの式
(Darcy-Weisbach Equation)

$$\Delta P = \lambda \frac{L}{D} \frac{\rho u^2}{2}$$

λ :管摩擦係数(Darcy’s friction factor)

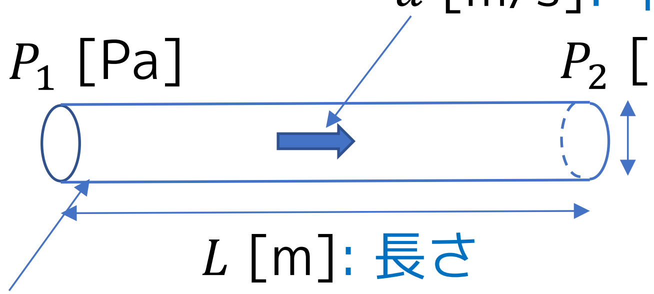
ρ[kg/m3]: 密度

μ[Pa.s]: 粘性率

$\Delta P = P_1 - P_2$: 圧力損失

u [m/s]: 平均流速

P_1 [Pa]　　P_2 [Pa]



D [m]: 内径

L [m]: 長さ

ε [m]: 表面粗さ(roughness)

$\Delta = \dfrac{\varepsilon}{D}$:相対粗さ　　$Re = \dfrac{\rho u D}{\mu}$:レイノルズ数

λ :管摩擦係数(Darcy's friction factor)

$Re < Re_1$ 層流

ハーゲン・ポアズイユ流
(Hagen-Poiseuille flow)

$$\lambda = \frac{64}{Re} \qquad u = \frac{\Delta P}{L}\frac{r^2}{8\mu}$$

$Re > Re_2$ 乱流

コールブルック・ホワイトの式
(Colebrook-White Equation)

$$\frac{1}{\sqrt{\lambda}} = -2\log_{10}\left(\frac{2.51}{Re\sqrt{\lambda}} + \frac{\varepsilon/D}{3.7}\right) = -2\log_{10}\left(\frac{2.51}{Re\sqrt{\lambda}} + 0.27\Delta\right)$$

$Re_1 < Re < Re_2$ 乱流と層流の遷移領域の範囲

$$Re_1 = 745\exp\left(if\ \Delta < 0.0065\ then\ 1\ else\ \frac{0.0065}{\Delta}\right)$$

[Samoilenko 1968; Idelchik 1994, p. 81, sect. 2.1.21]

$$Re_2 = Re_{turbulent} = 4000$$

$\lambda_2 \equiv \lambda Re^2$ の導入

層流の場合

$$\lambda = \frac{64}{Re} \quad \Leftrightarrow \quad Re = \frac{\lambda_2}{64} \quad \text{または} \quad \lambda_2 = 64\, Re$$

$Re \to 0$ で λ は発散するが λ_2 は発散しない。

乱流の場合

$$\frac{1}{\sqrt{\lambda}} = -2\log_{10}\left(\frac{2.51}{Re\sqrt{\lambda}} + 0.27\Delta\right) \quad \Leftrightarrow Re = -2\sqrt{\lambda_2}\log_{10}\left(\frac{2.51}{\sqrt{\lambda_2}} + 0.27\Delta\right)$$

コールブルック・ホワイトの式が陰関数でなくなる。

圧力と λ_2 の関係

$$\Delta P = \lambda \frac{L}{D}\frac{\rho u^2}{2}, \quad Re = \frac{\rho u D}{\mu} \quad \Leftrightarrow \quad \lambda_2 = |\Delta p|\frac{2D^3\rho}{L\mu^2} \quad \text{または} \quad \Delta p = \pm\frac{L\mu^2}{2D^3\rho}\lambda_2$$

質量流量とレイノルズ数の関係

$$Re = \frac{\rho u D}{\mu}, \qquad \dot{m} = \rho u A \qquad \Leftrightarrow \dot{m} = \frac{\mu A}{D} Re \quad \text{または} \quad Re = \frac{D}{\mu A}|\dot{m}|$$

massFlowRate_dp　圧力差から質量流量を求める

① $\lambda_2 = |\Delta p| \dfrac{2D^3\rho}{L\mu^2}$

② $Re = \dfrac{\lambda_2}{64}$　まず、層流を仮定してレイノルズ数を求める。

$Re > Re_1$ なら

$$Re = -2\sqrt{\lambda_2}\log_{10}\left(\frac{2.51}{\sqrt{\lambda_2}} + 0.27\Delta\right)$$

層流の範囲外なら乱流を
仮定してレイノルズ数を求める。

$Re < Re_2$ なら　遷移領域で、補間関数でレイノルズ数を求める。

$$Re = interplateRegion2(Re, Re_1, Re_2, \Delta, \lambda_2)$$

③ $\dot{m} = \pm\dfrac{\mu A}{D}Re$　レイノルズ数から質量流量を計算する。
符号はΔpに合わせる。

pressureLoss_m_flow 質量流量から圧力差を求める

スワミー・ジャインの式(Swamee-Jain equation)
（コールブルック・ホワイトの式の近似式）

$$\lambda = 0.25\left[\log\left\{\frac{\Delta}{3.7}+\frac{5.74}{Re^{0.9}}\right\}\right]^{-2}$$

① $Re = \dfrac{D}{\mu A}|\dot{m}|$

② $Re \leq Re_1$ なら

$$\lambda_2 = 64\,Re$$

$Re \geq Re_2$なら

$$\lambda_2 = 0.25\left(\frac{Re}{\log\left\{\frac{\Delta}{3.7}+\frac{5.74}{Re^{0.9}}\right\}}\right)^2$$

$Re_1 < Re < Re_2$ なら　遷移領域の補間関数

$$\lambda_2 = interplateInRegion2(Re, Re1, Re2, \Delta)$$

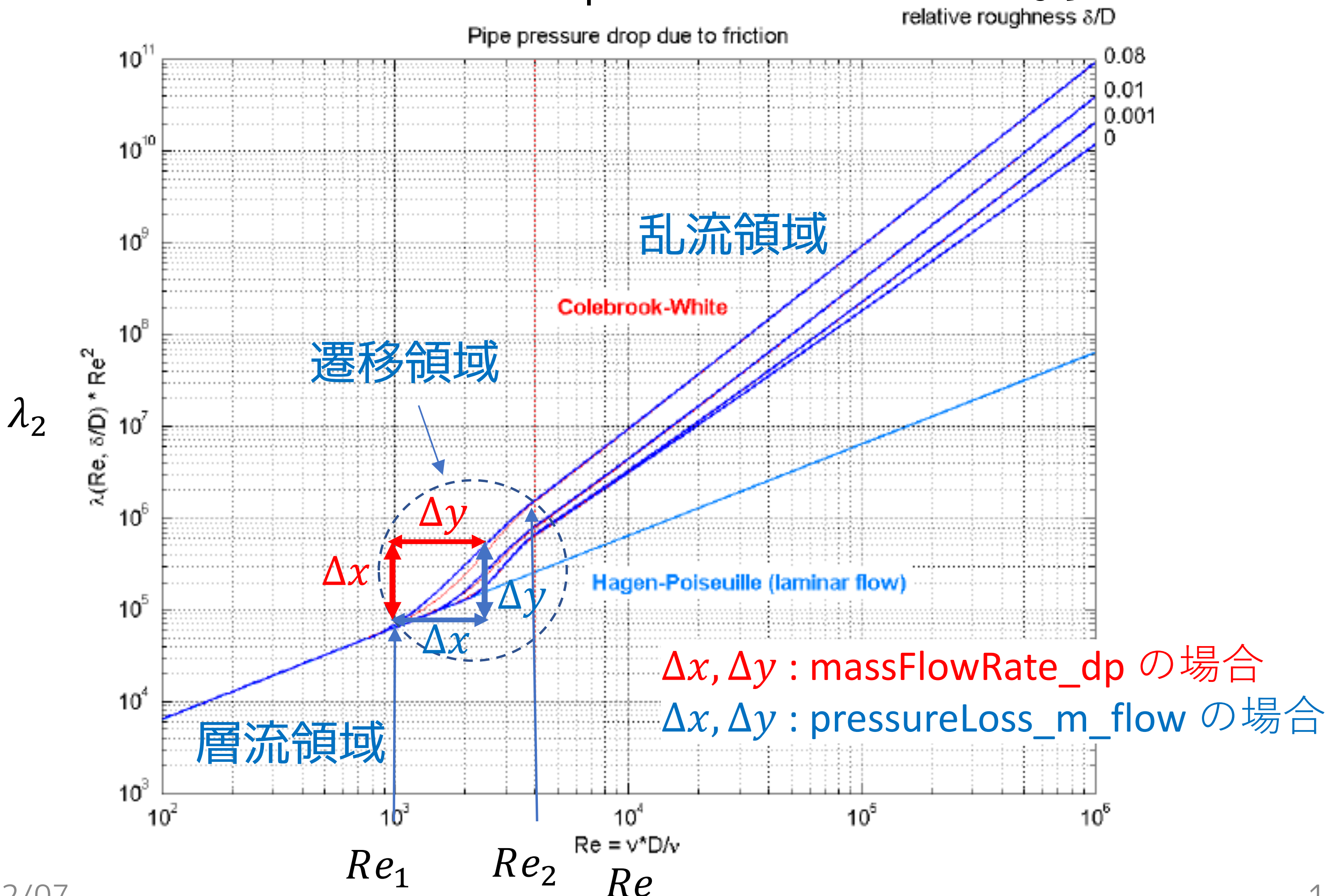
③ $\Delta p = \pm\dfrac{L\mu^2}{2D^3\rho}\lambda_2$

λ_2　から圧力差を計算する。
符号は$\dot{m}$に合わせる。

遷移領域の補間方法の考え方

対数グラフ上の距離 $\Delta x, \Delta y$ を用い、Δy を Δx の3次多項式で近似し、Re_1 と Re_2 で、それぞれ層流領域の曲線と乱流領域の曲線に、１階微分まで連続になるようにつなぐ。

Modelica.Fluid.UsersGuid.ComponentDefinition.WallFrincion より



付録２. PartialLumpedVesselの圧力計算式

(1) 一般的な圧力損失係数と圧力流量関係式
(2) 液位がポートの高さと近い場合の補正 penetrationの効果
(3) 流量が小さい場合の regularization (流れの適正化)
(4) ホモトピー法による非線形方程式の計算

(1) 一般的な圧力損失係数と圧力流量関係式

流入

$$p_{in} + \frac{\dot{m}^2}{2\rho_{in}A_{in}^2} - \zeta_{in} \cdot \frac{\dot{m}^2}{2\rho_{in}A_{in}^2} = p_{vessel} + \frac{\dot{m}^2}{2\rho_{vessel}A_{vessel}^2}$$

静圧　動圧　圧力損失　静圧　動圧

$$\Rightarrow \quad p_{in} = p_{vessel} + \left(\zeta_{in} - 1 + \frac{A_{in}^2}{A_{vessel}^2}\right)\frac{1}{2\rho_{in}A_{in}^2} \cdot \dot{m}^2$$

流出

$$p_{vessel} + \frac{\dot{m}^2}{2\rho_{vessel}A_{vessel}^2} - \zeta_{out} \cdot \frac{\dot{m}^2}{2\rho_{out}A_{out}^2} = p_{out} + \frac{\dot{m}^2}{2\rho_{out}A_{out}^2}$$

静圧　動圧　圧力損失　静圧　動圧

$$\Rightarrow \quad p_{out} = p_{vessel} - \left(\zeta_{out} + 1 - \frac{A_{out}^2}{A_{vessel}^2}\right)\frac{1}{2\rho_{vessel}A_{out}^2}\dot{m}^2$$

動圧 $\frac{1}{2}\rho u^2 = \frac{(\rho u A)^2}{2\rho A^2} = \frac{\dot{m}^2}{2\rho A^2}$ (単位体積あたりの運動エネルギー)

(2)液位がポートの高さに近い場合の補正

容器へ流入する時の ($\dot{m}_{port} > \dot{m}_{turbulent}$)**のポートの静圧**

$$p_{in} = p_{vessel_i} + \underbrace{\frac{1}{2A_i^2}\left(\zeta_{in_i} - 1 + \frac{A_i^2}{A_{vessel}^2}\right)\frac{1}{\rho_{in_i}}\color{red}{penetration_i}}_{k_1} \cdot m_i^2$$

容器から流出する時の ($\dot{m}_{port} < -\dot{m}_{turbulent}$)**のポートの静圧**

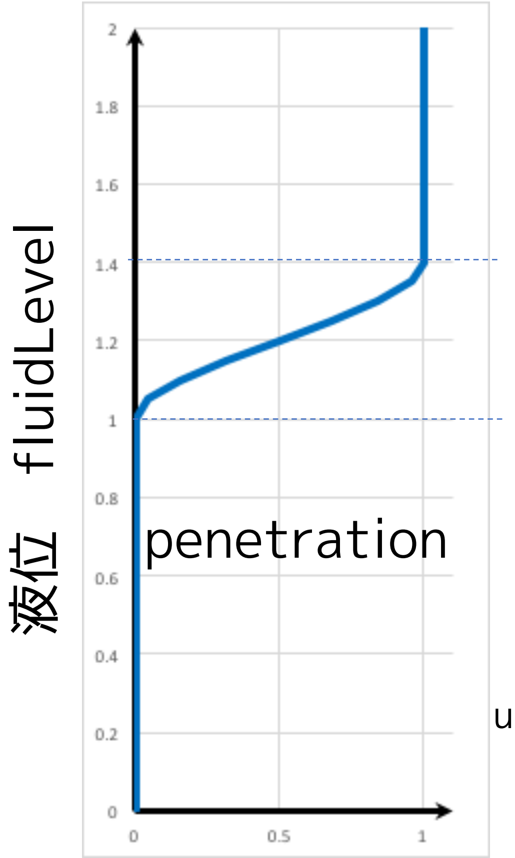
$$p_{out} = p_{vessel_i} - \underbrace{\frac{1}{2A_i^2}\left(\zeta_{out_i} + 1 - \frac{A_i^2}{A_{vessel}^2}\right)\frac{1}{\rho_{vessel}}\frac{1}{\color{red}{penetration_i}}}_{k_2} \cdot \dot{m}_i{}^2$$

液位がポート高さ+0.2D以下になると、 $penetration_i < 1$ となる。
流入時の圧力損失が小さくなり、流出時の圧力損失が大きくなる。

penetration の計算式

$$penetration_i = \boldsymbol{regStep}(fluidLevel - height_i - 0.1D_i, 1, 1\times10^{-3}, 0.1D_i)$$

```
ports_penetration[i]
= Utilities.regStep(fluidLevel - portsData_height[i] - 0.1*portsData_diameter[i],
  1, 1e-3, 0.1*portsData_diameter[i]);
```



penetration=1.0

ポート高さ+0.2D

ポート高さ

penetration=0.001

use_portsData=false のとき $penetration_i = 1$

Modelica.Fluid.Utilities.regStep
ステップ関数を連続で微分可能な曲線で近似する関数

$$y = \begin{cases} y1, & x > 0 \\ y2, & x \le 0 \end{cases} \Rightarrow y = \begin{cases} y1, & x > x_{small} \\ y2, & x < -x_{small} \\ f(y1, y2) & -x_{small} \le x \le x_{small} \end{cases}$$

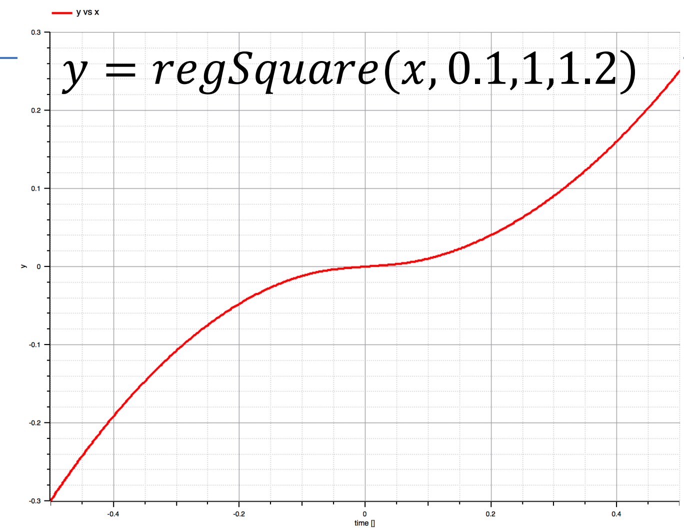
$-x_{small} < x < x_{small}$の領域は、$y1$から$y2$に変化する2次多項式 $f(x)$ で近似する。

```
y := smooth(1, if x >  x_small then y1 else
               if x < -x_small then y2 else
               if x_small > 0 then
                 (x/x_small)*((x/x_small)^2 - 3)*(y2-y1)/4 + (y1+y2)/2
               else
                 (y1+y2)/2);
```

(3) 流量が小さい場合の regularization (流れの適正化)

$$p_{port} = p_{vessel_i} + \frac{1}{2A_i}\mathrm{regSqure2}(\dot{m}, \dot{m}_{turbulent}, k_1, k_2)$$

p_{in} とp_{out} を$|\dot{m}_i| < \dot{m}_{turbulent_i}$のときに滑らかにつなぐ



Modelica.Fluid.Utilities.regSquare2

$$y = \mathrm{regSquare}\left(x, x_{small}, k_1, k_2, use_{yd0}, yd0\right)$$

不連続な因数を持つ非対称な正方（自乗）

$$y = \begin{cases} k_1 x^2, & x \geq 0, k_1 > 0 \\ -k_2 x^2, & x < 0, k_2 > 0 \end{cases}$$

を $-x_{small} \leq x \leq x_{small}$ で滑らかに繋ぐ。

- $-x_{small} \leq x \leq 0$ と $0 \leq x \leq x_{small}$ を、それぞれ別々の3次多項式で近似する。
- $x = 0$で微分がゼロにならない。（逆数が無限大にならない。）
- 全領域で一階微分が連続になる。
- $use\_yd0 = \mathrm{false}$ （デフォルト）なら2つ3次多項式の二階微分が $x = 0$ で一致する。
- $use\_yd0 = \mathrm{true}$ なら、$x = 0$における一階微分 $yd0$ を引数で指定する。

(4) ホモトピー法による非線形方程式の計算

流量と圧力損失の関係が非線形となるのでホモトピーオペレータを使用する。

actual model

$$p_{port} = p_{vessel} + \frac{1}{2A^2}\mathrm{regSqure2}(\dot{m}, \dot{m}_{turbulent}, k_1, k_2)$$

simplified model

$$p_{port} = p_{vessel}$$

homotopy model

ホモトピーオペレータ

$$p_{port} = homotpy\left(p_{vessel} + \frac{1}{2A^2}regSqure2(\dot{m}, \dot{m}_{turbulent}, k_1, k_2), p_{vessel}\right)$$

PartialVessel の圧力計算式

```
// fluid flow through ports
  regularFlow[i] = fluidLevel >= portsData_height[i];
  inFlow[i]      = not regularFlow[i] and (s[i] > 0 or portsData_height[i] >= fluidLevel_max);
  if regularFlow[i] then
    // regular operation: fluidLevel is above ports[i]
    // Note: >= covers default values of zero as well
    if use_portsData then
      /* Without regularization
         ports[i].p = vessel_ps_static[i] + 0.5*ports[i].m_flow^2/portAreas[i]^2
                      * noEvent(if ports[i].m_flow>0 then zeta_in[i]/portInDensities[i] else -zeta_out[i]/medium.d);
      */

      ports[i].p = homotopy(vessel_ps_static[i] + (0.5/portAreas[i]^2*Utilities.regSquare2(ports[i].m_flow, m_flow_turbulent[i],
                           (portsData_zeta_in[i] - 1 + portAreas[i]^2/vesselArea^2)/portInDensities[i]*ports_penetration[i],
                           (portsData_zeta_out[i] + 1 - portAreas[i]^2/vesselArea^2)/medium.d/ports_penetration[i])),
                    vessel_ps_static[i]);
      /*
        // alternative formulation m_flow=f(dp); not allowing the ideal portsData_zeta_in[i]=1 though
        ports[i].m_flow = smooth(2, portAreas[i]*Utilities.regRoot2(ports[i].p - vessel_ps_static[i], dp_small,
                                 2*portInDensities[i]/portsData_zeta_in[i],
                                 2*medium.d/portsData_zeta_out[i]));
      */
    else
      ports[i].p = vessel_ps_static[i];
    end if;
    s[i] = fluidLevel - portsData_height[i];
```

ホモトピーオペレータ

regularization

k_1

k_2

penetration