OpenModelica超初級チュートリアル

7.5 番外編　stream変数

注意事項

- 本チュートリアルは以下の内容が理解できていることを前提としております。
  「OpenModelica超初級チュートリアル1.解析モデルの作成と実行」
  「OpenModelica超初級チュートリアル2.コーディング」
  「OpenModelica超初級チュートリアル3.モデルのカスタマイズ1」
  「OpenModelica超初級チュートリアル4.モデルのカスタマイズ2」
  「OpenModelica超初級チュートリアル5.モデルのカスタマイズ3」
  「OpenModelica超初級チュートリアル7.プラントモデルの作り方」

- OpenModelica1.14.1 (64bit – windows版)を利用して本チュートリアルは作成されています。

目次

目次

流体の移動現象を表す物理量

流体の移動現象を表す場合、圧力、質量流量、比エンタルピー*1や化学成分の質量分率を取り扱うことがあります。
この中で比エンタルピーや質量分率は流れに応じて輸送される単位質量や単位体積当たりの物理量です。
(これらを便宜的に輸送される比状態量と呼ぶことにします)
輸送される比状態量をstream変数で表すと計算が安定しコードもシンプルになります。

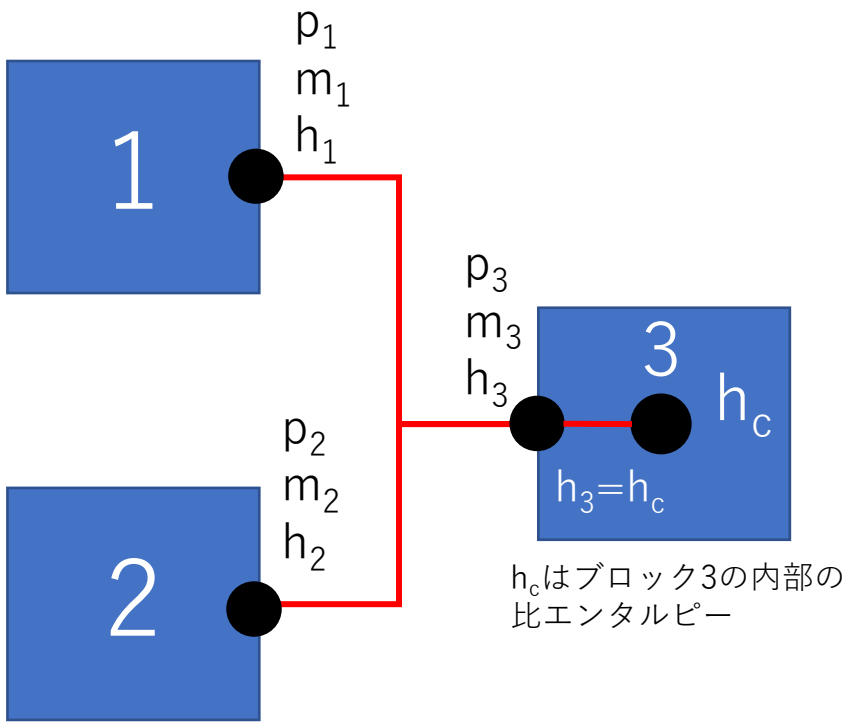
流体の移動現象を表す為にconnectorクラス内で宣言する物理量の例

変数の種類	表現する対象	例となる物理量	Modelicaコード例	単位
across変数	ポテンシャル	圧力	Real p;	N/m^2
flow変数	フラックス	質量流量	flow Real m;	kg/s
stream変数	輸送される比状態量	比エンタルピー 質量分率	stream Real h; stream Real Xi;	J/kg kg/kg

*1　単位質量当たりのエンタルピーH*2
*2　熱エネルギーを表す内部エネルギーと圧力pと体積Vに関するエネルギーを表すpVの和(H=U+pV)
　　流体の熱輸送現象を計算する際にエンタルピーがよく使用される

stream変数に関する計算式

across変数、flow変数はポート接続時に自動的に方程式が作成されます。
しかしstream変数は方程式は自動的に作成されません。そのためstream変数を使ってモデルを作成する場合、計算したい対象に合わせてオペレータを選択する必要があります。



hcはブロック3の内部の比エンタルピー

3つのブロックの接続と方程式

下付き添え字が数字の時、各ブロックのポートの値

across変数 p

$$p_1 = p_2 = p_3$$

flow変数 m

$$m_1 + m_2 + m_3 = 0$$

本資料での記号は以下を表します

圧力	p[N/m²]
質量流量	m[kg/s]
比エンタルピー	h[J/kg]
エンタルピー流量	H[J/s]
正の微小量	ϵ

stream変数 h

ブロック3のポートのh_3を例示 (j = 1,2)

$$actualStream(h_3) = \begin{cases} \dfrac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} & (m_3 > 0) \quad \text{流入} \\ h_3 & (m_3 \le 0) \quad \text{流出} \end{cases} \quad \cdots (1)$$

or

$$inStream(h_3) = \frac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} \quad \cdots (2)$$

計算したい内容に合わせてどちらかを選択

本資料では輸送される比状態量が式(1)や式(2)で表されることを確認していきます。

stream変数の難しさ

stream変数は以下の5つの理由から難しいと感じてしまうかもしれません。
本資料では、一つずつ整理しながら出来るだけ分かりやすく解説していきます。

stream変数が表す物理現象

- stream変数が表す代表的な物理現象に比エンタルピーや化学成分の質量分率があります。それらの性質を解説します。

ゼロ点の回避

- 数値計算上の不安定性の原因となるゼロ点の発生を回避するための工夫を紹介します。

双方向流れ(逆流)への対応

- 輸送される比状態量は流れの方向に応じて計算式を切り替える必要があります。

分岐・合流

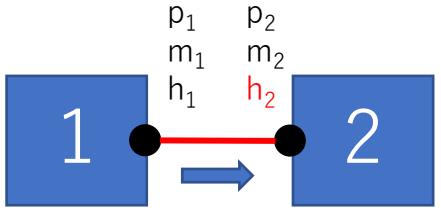
- 分岐・合流時の計算は一見複雑ですが保存式から考えるとシンプルなものとなります。

実装時のルールとオペレータ

- stream変数はinStreamもしくはactualStreamオペレータと必ずセットで使用します。
- 使い方や注意点についても解説します。

stream変数の計算式サマリー

本資料では、左図のフローチャートに従ってstream変数の解説を行います



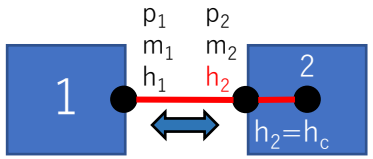
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

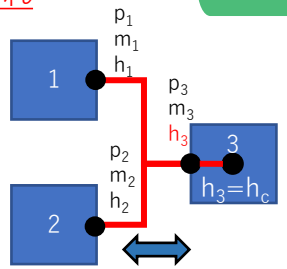
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

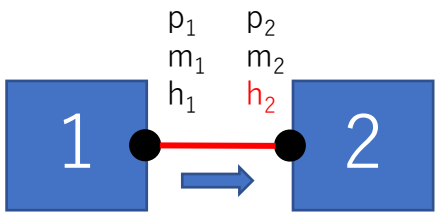
or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

目次

stream変数が表す物理現象と計算式

本資料では、左図のフローチャートに従ってstream変数の解説を行います



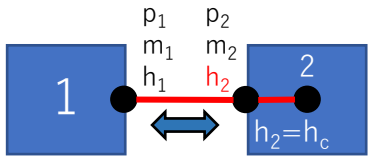
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

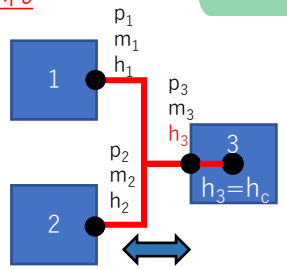
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

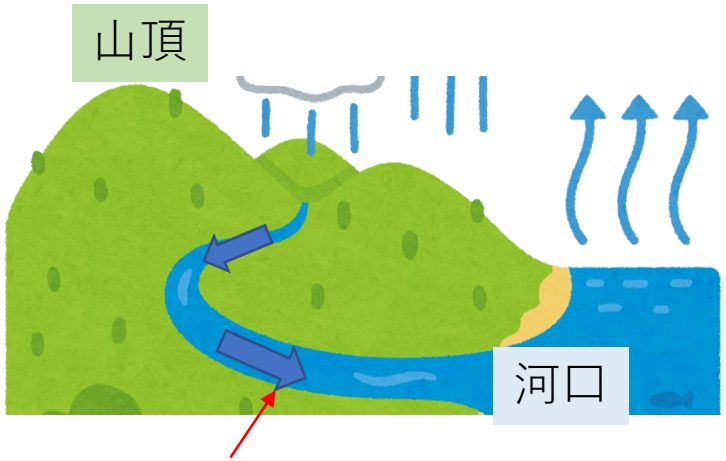
$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

stream変数が表す物理現象

物理現象の中には流動量(flow変数)に応じて輸送される物理量(輸送量)があります。
Modelicaでは、単位質量や体積当たりの輸送量をstream変数で定義すると色々と便利です。



輸送量　：　流れに応じて移動する物理量(ベクトル)
例：エンタルピー、物質量

輸送される比状態量(stream変数)：単位質量や単位体積当たりに含まれる輸送量の大きさ(スカラー)
例：比エンタルピー、質量分率

輸送量とstream変数の関係は次式となります

$$\text{(輸送量)} = \text{(flow変数)} \times \text{(stream変数)}$$

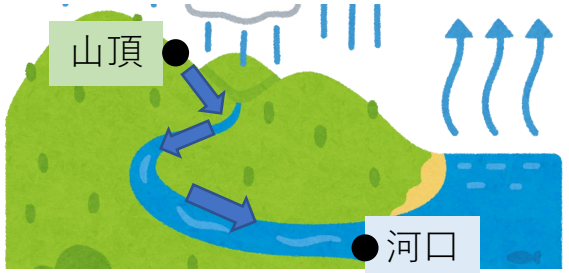
stream変数を比エンタルピーとすると、エンタルピー流量の計算は次式となる。

$$H = m \times h$$

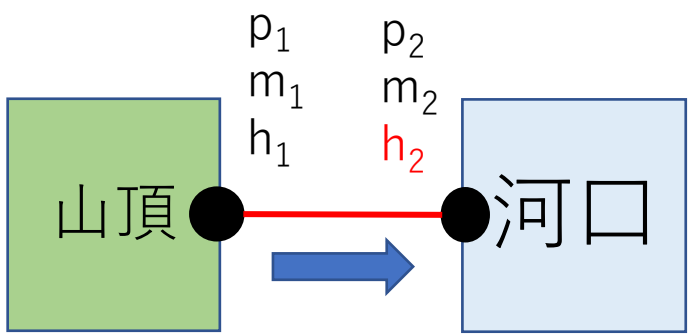
質量流量	m[kg/s]
比エンタルピー	h[J/kg]
エンタルピー流量	H[J/s]

stream変数の計算式

川の流れをブロック図で表して
山頂から河口に輸送される比エンタルピーh$_2$を求めてみましょう。



川の流れをブロック図化



川の流れのブロック図

質量流量の保存式

$$m_1 + m_2 = 0$$

$$m_2 = -m_1$$

エンタルピーの保存式からh$_2$を導出

$$H_1 + H_2 = 0$$

$$m_1 h_1 + m_2 h_2 = 0$$

h$_2$について整理

$$h_2 = \frac{-m_1 h_1}{m_2}$$

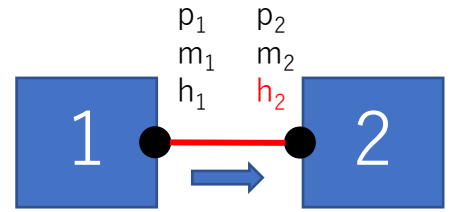
m$_2$=-m$_1$を代入

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

目次

ゼロ点の回避

本資料では、左図のフローチャートに
従ってstream変数の解説を行います



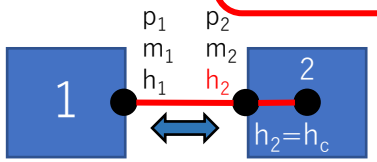
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

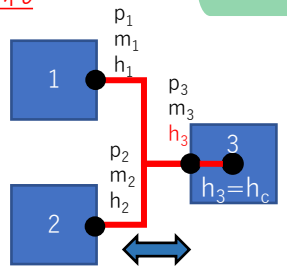
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

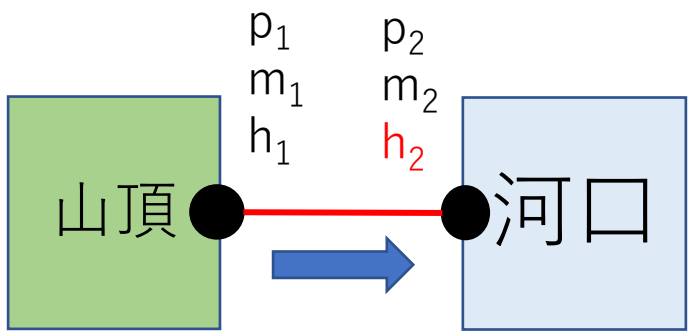
$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

ゼロ点の回避

逆流が発生したり流れが止まると流量が0となる瞬間が発生します。
数値計算において、0が生じると計算がゼロ割りや非明示な解がなくなるなど不安定になることが多くなります。
0を避けるため、 流量は以下のように計算することにしましょう。



質量流量がゼロとならないようにした場合のh_2の計算

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

m_1が0に近づいたらεに切り替え

$$h_2 = \frac{\max(-m_1, \epsilon) h_1}{\max(-m_1, \epsilon)}$$

流入の式はΣ以外導出できました

(正の微小量 ϵ)

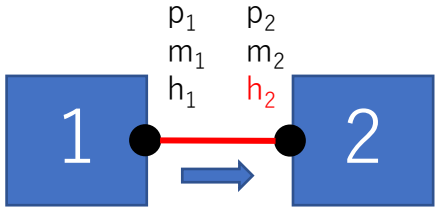
比較のためacturalStreamの計算式を再掲

$$actualStream(h_3) = \begin{cases} \dfrac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} & (m_3 > 0) \quad \text{流入} \\ h_3 & (m_3 \leq 0) \quad \text{流出} \end{cases} \quad \cdots (2)$$

目次

双方向流れ(逆流)への対応

本資料では、左図のフローチャートに
従ってstream変数の解説を行います



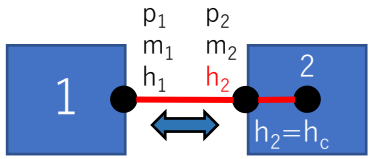
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

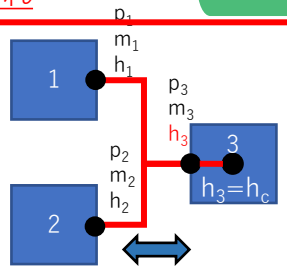
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

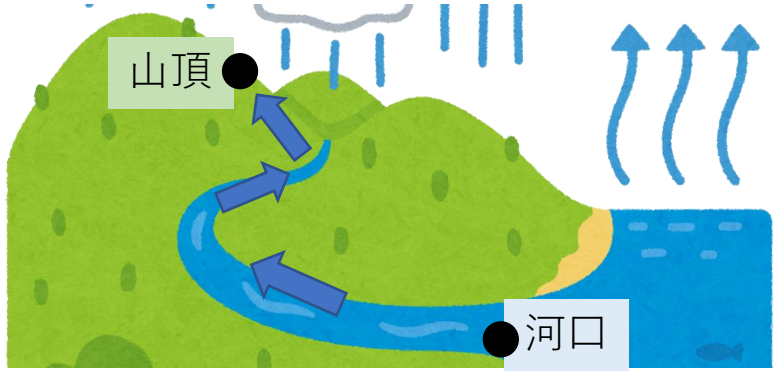
or

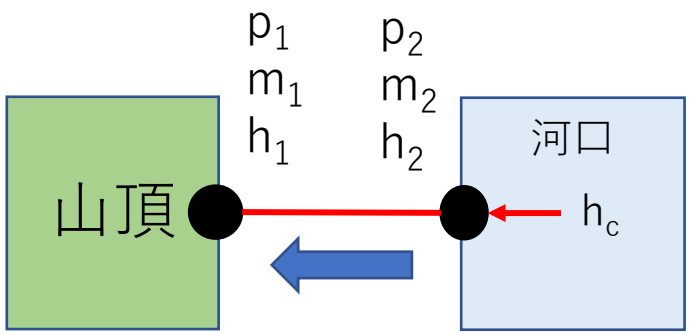
$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

双方向流れ(逆流)

海嘯(かいしょう)と呼ばれる現象によってアマゾン川や銭塘江(せんとうこう)では逆流が発生します。
逆流が起きた場合でも計算が出来るようにstream変数の計算を考えましょう。

川の流れが逆流(河口→山頂への流れ)となった場合、
河口モデルのポートの比エンタルピーh_2は、河口より上流側(海)の比エンタルピーh_cを仮定することで計算できます。 h_cは河口モデルの内部に定義します。





逆流(河口→山頂への流れ)の場合
h2の計算式は次式となる。

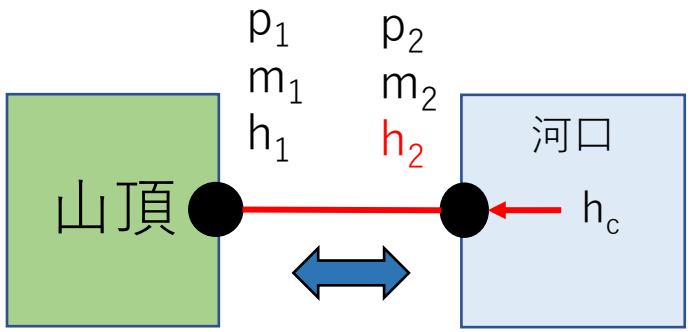
$$h_2 = h_c$$

stream変数は上流側の値を定義することで逆流も計算可能です。

逆流を考慮した計算式

逆流した場合にも対応できるように計算式を考えてみましょう。
河口のポートの比エンタルピーh_2の計算式は次のように場合分けが必要です。

$$h_2 = \begin{cases} \dfrac{\max(-m_1, \epsilon) h_1}{\max(-m_1, \epsilon)} & (\text{山頂から流れてくる場合 } m_2>0) \\ h_c & (\text{河口から流れてくる場合 } m_2<0) \end{cases}$$



比較のためacturalStreamの計算式を再掲

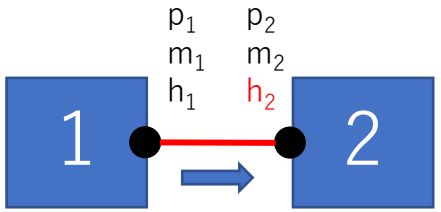
$$actualStream(h_3) = \begin{cases} \dfrac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} & (m_3 > 0) \\ h_c & (m_3 \le 0) \end{cases} \quad \cdots (2)$$

流れの方向に応じて式を切り替えることで
上の式と式(2)は似た形式となってきました。

目次

分岐・合流

本資料では、左図のフローチャートに
従ってstream変数の解説を行います



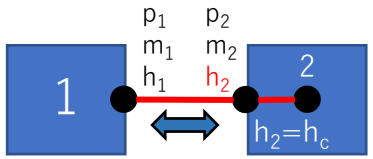
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

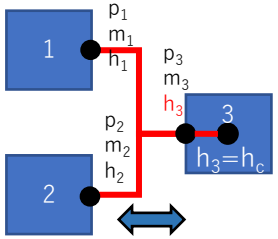
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

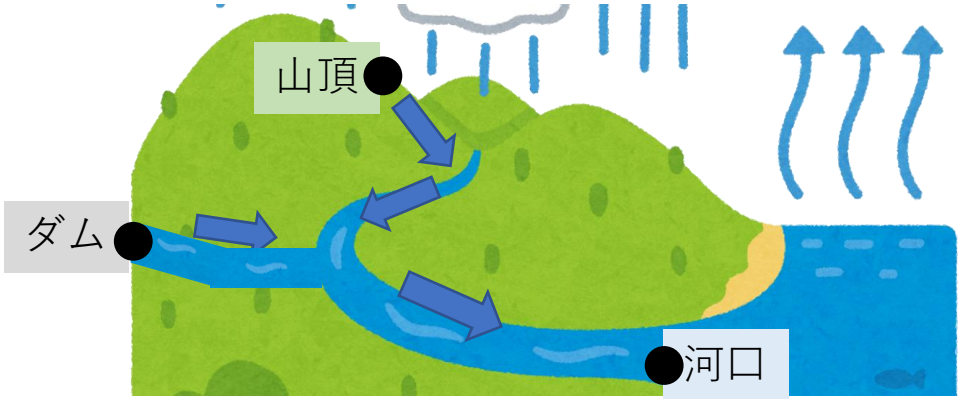
$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

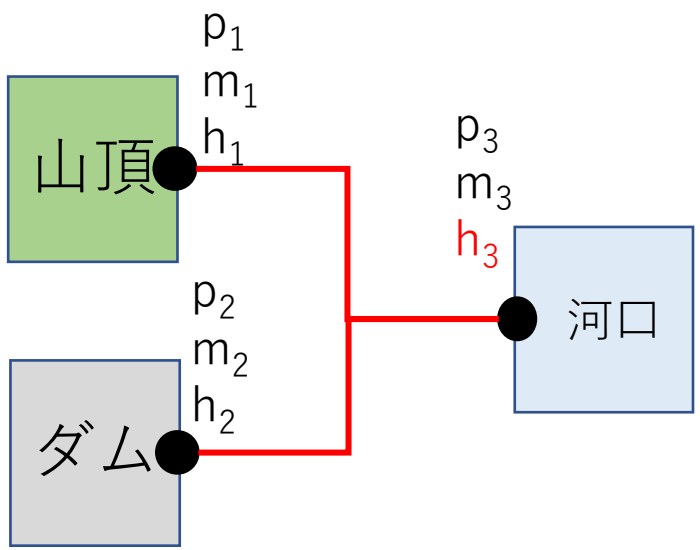
or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

分岐・合流

以下の図のように山頂とダムの水が合流して河口モデルへ流れるとしましょう。
このときの河口のポートの比エンタルピーh$_3$を求めてみましょう。





合流部のエンタルピーの保存式からh$_3$を導出

$$H_1 + H_2 + H_3 = 0$$

⇒

$$m_1 h_1 + m_2 h_2 + m_3 h_3 = 0$$

⇒

$$h_3 = \frac{-m_1 h_1 - m_2 h_2}{(-m_1 - m_2)}$$

⇒

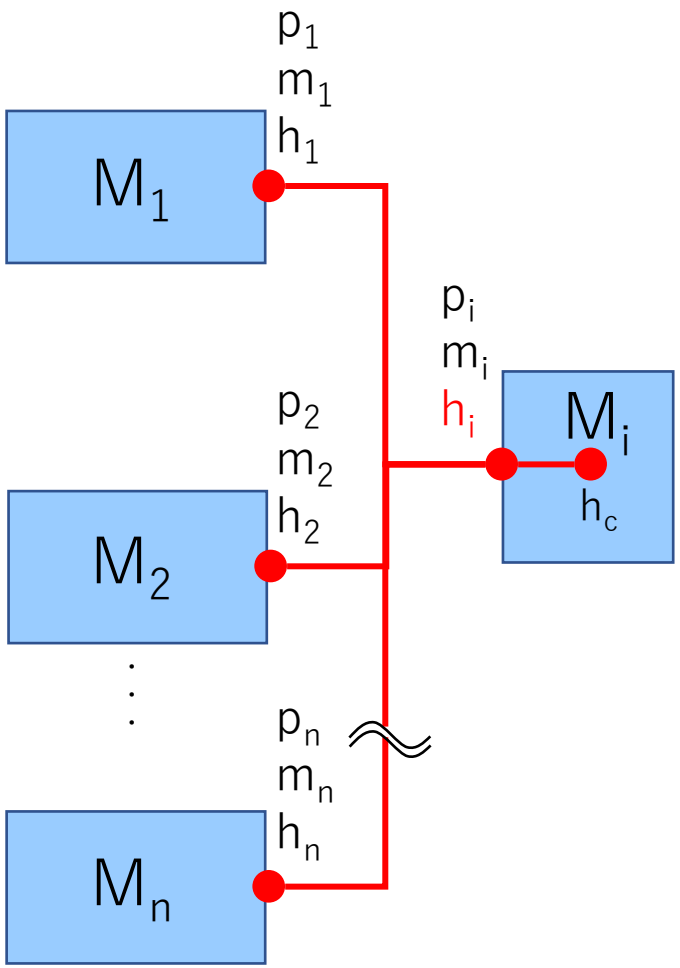
$$h_3 = \frac{\sum_{j=1,2} -m_j h_j}{\sum_{j=1,2} -m_j}$$

ゼロ点回避 ⇒

$$h_3 = \frac{\sum_{j=1,2} \max(-m_j, \epsilon) h_j}{\sum_{j=1,2} \max(-m_j, \epsilon)}$$

分岐・合流の計算の一般化

下図のように接続した場合、モデルM_iのポートのstream変数h_iは次式で計算できます。



M_iへ流入する場合$(m_i > 0)$

$$h_i = \frac{\sum_j -m_j h_j}{\sum_j -m_j} \qquad (j \neq i)$$

ゼロ点回避

$$h_i = \frac{\sum_j \max\left(-m_j, \epsilon\right) h_j}{\sum_j \max(-m_j, \epsilon)} \qquad (j \neq i)$$

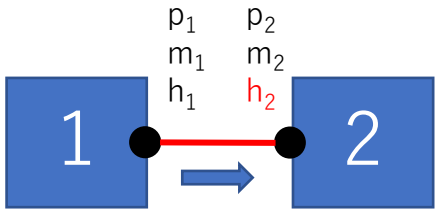
M_iから流出する場合$(m_i < 0)$

$$h_i = h_c$$

目次

実装時のルールとオペレータ

本資料では、左図のフローチャートに従ってstream変数の解説を行います



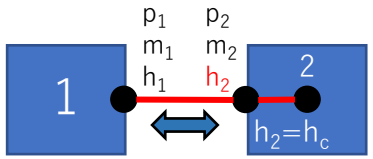
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

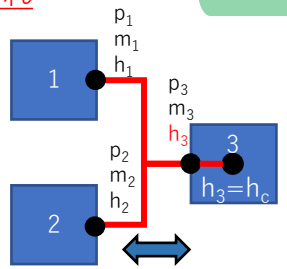
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

実装時のルールとオペレータ

stream変数には以下のルールがあります

No.	ルール
1	接頭辞streamはコネクタークラス内のみで使用できます。
2	stream変数が宣言されたコネクタークラスはacross変数とflow変数が一つずつ必要です。
3	stream変数は、ポート同士が接続されても自動的に方程式は生成されません。 そのためinStreamオペレータあるいはactualStreamオペレータを使用してstream変数に関する値を計算する必要があります。

二つのオペレータ

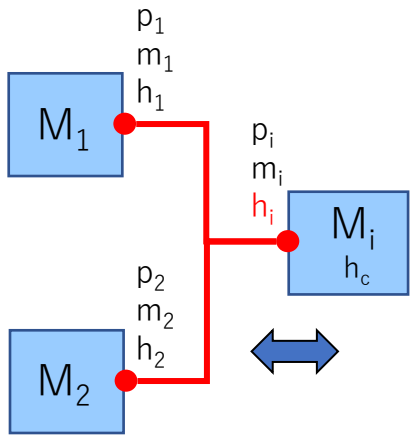
actualStreamオペレータ

ポートのstream変数を引数として、逆流を考慮してそのポートを通過する輸送される比状態量の値を返します。actualStreamオペレータは本当に必要な時のみ使用してください。

$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

M_iモデルへ流入する場合

M_iモデルから流出する場合
h_iの値は別途定義しておく



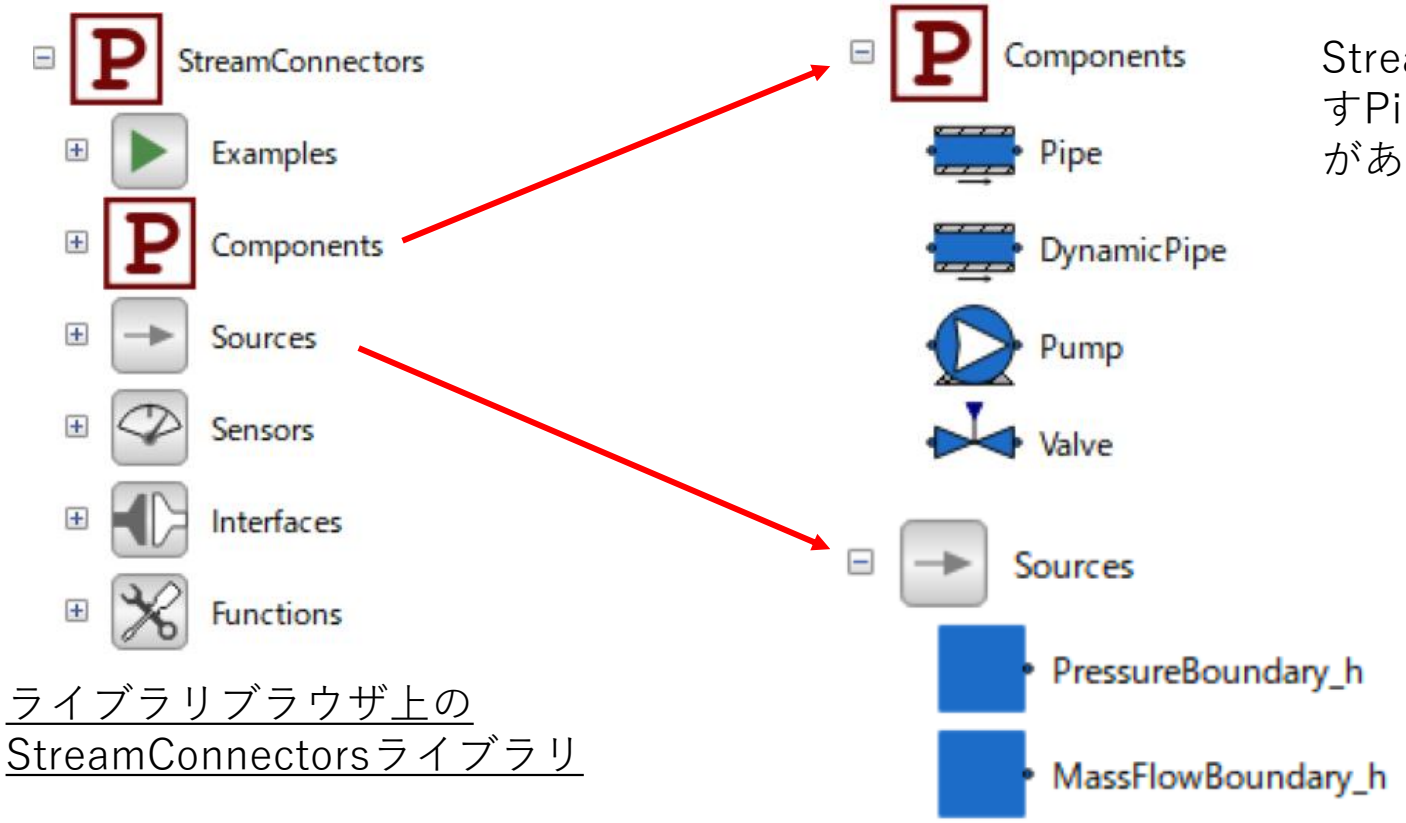
inStreamオペレータ

ポートのstream変数を引数として、流れの向きとは関係なくそのポートに接続されたコネクタから計算される輸送される比状態量の値を返します。流れの向きとは関係なく計算されるため使用には注意が必要ですがactualStreamより計算負荷が少なく安定的です。

$$inStream(h_i) = \frac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} \qquad (j \neq i)$$

stream変数学習用ライブラリ StreamConnectors

stream変数を学習するためのライブラリStreamConnectorsがOpenModelicaのシステムライブラリにあります。
(ソースコードはRene Just Nielsen氏(アカウント名)のGitHubで公開されています。)
このライブラリ内を参考にstream変数の実装方法を確認しましょう。



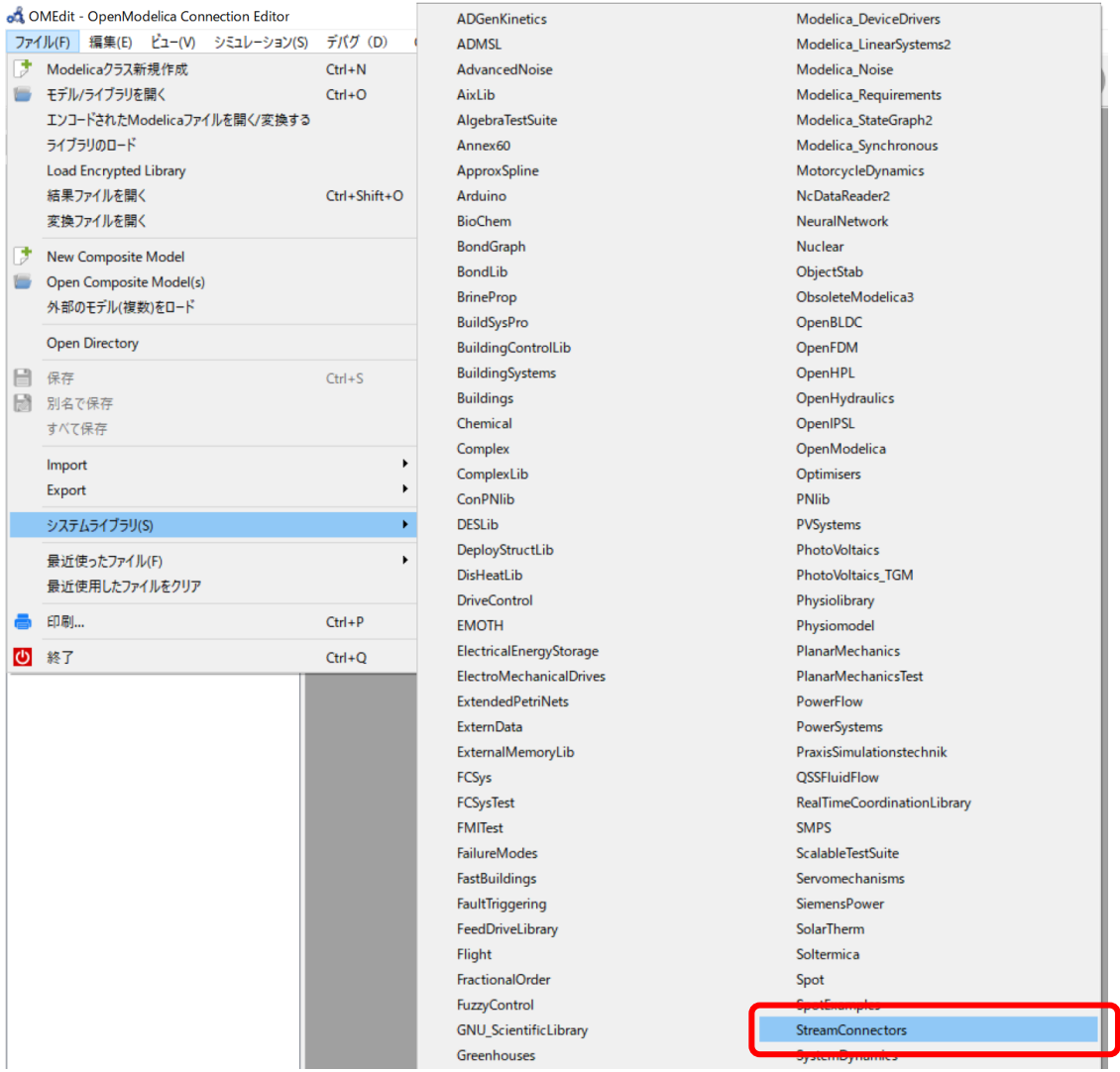
ライブラリブラウザ上の
StreamConnectorsライブラリ

StreamConnectorsライブラリには流体抵抗を表すPipeモデル、DynamicPipeやPumpモデルなどがあります。

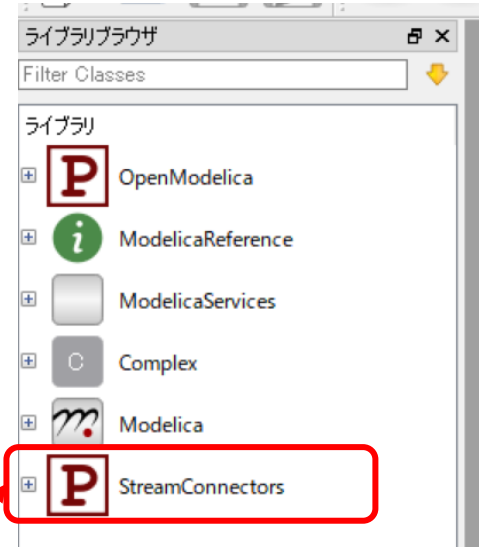
境界条件として以下のモデルがあります。
・圧力、比エンタルピー定義
・質量流量、比エンタルピー定義

StreamConnectorsライブラリのローディング

1. StreamConnectorsライブラリはOpenModelicaの「ファイル」→「システムライブラリ」→「 StreamConnectors 」から開くことができます。

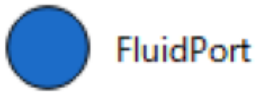


2. ライブラリブラウザから確認できます。



connectorクラスの実装例

StreamConnectorsライブラリでは、以下のようにconnectorクラスFluidPortに圧力、質量流量、比エンタルピーが宣言されています。




```
connector FluidPort "Simple stream connector.“
  Real p "Potential/effort variable";
  flow Real m_flow "Flow variable";
  stream Real h_outflow "Specific enthalpy";
end FluidPort;
```

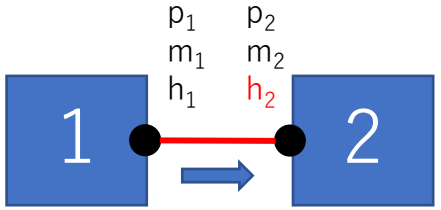
stream変数のルール1,2に基づいてconnectorクラスが作成されています

No.	ルール
1	接頭辞streamはコネクタークラス内のみで使用できます。
2	stream変数が宣言されたコネクタークラスはacross変数とflow変数が一つずつ必要です。

目次

actualStreamオペレータの実装例

本資料では、左図のフローチャートに従ってstream変数の解説を行います



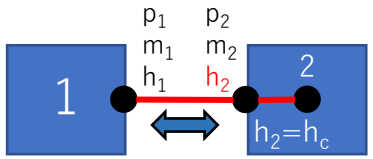
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

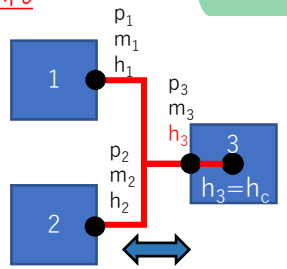
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

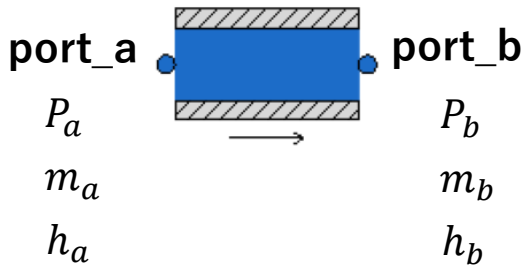
$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

actualStreamオペレータの実装例 – Pipeモデル

Pipeクラスを参考にactualStreamオペレータの使用方法を確認しましょう。
まずは系の途中で熱の流入出がない場合を想定します。



StreamConnectors.Components.Pipe

可読性のため以下のように置き換えています。

P_X：port_X.p
m_X：port_X.m_flow
h_X：port_X.h_outflow
X：ポート名

Pipeクラスの概要

- 両端のFluidPort(port_a, port_b)から圧力p、質量流量m、比エンタルピーhが流入出する単成分の流体抵抗モデル

実装式(簡易的に記載しています)

質量保存則

$$m_a + m_b = 0$$

圧力-質量流量の関係式

$$P_a - P_b = Km_a^2$$ (パラメータ K : pressure drop coeficient)

エンタルピーの保存式

流入出エンタルピーの保存

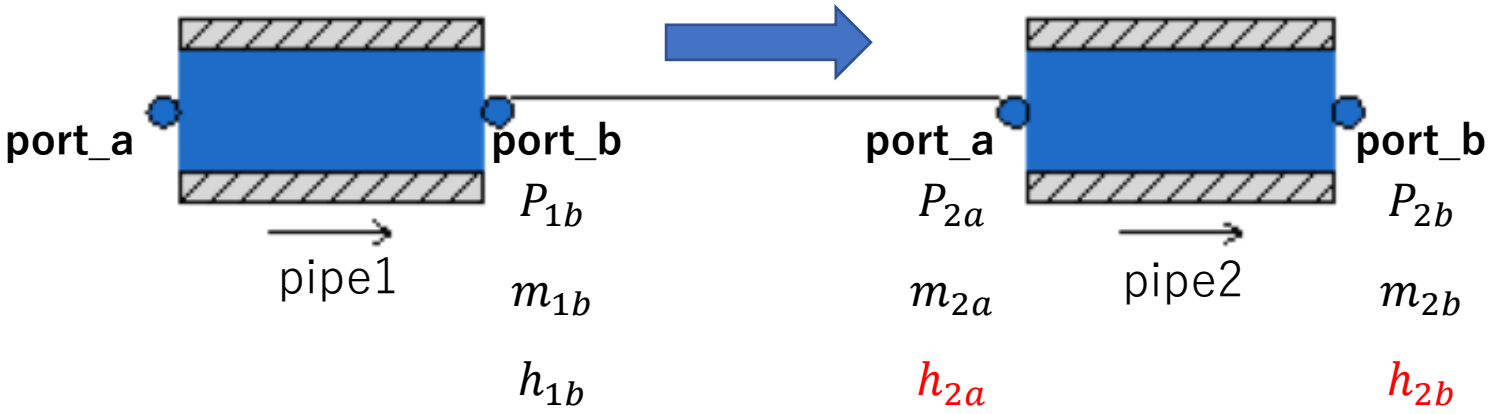
$$m_a \times \mathrm{actualStream}(h_a) + m_b \times \mathrm{actualStream}(h_b) = 0$$

流入出する比エンタルピーは等しい

$$h_a = h_b$$

actualStreamオペレータの実装例 – 簡単な例

下図のような場合にpipe2のh_{2a}, h_{2b}に着目してactualStreamオペレータの計算順序を確認しましょう。通過する比エンタルピーはすべて一定のため$h_{1b}=h_{2a}=h_{2b}$となりますが、計算になれるために1つずつ計算してみます。



赤字：未知変数
黒字：既知変数

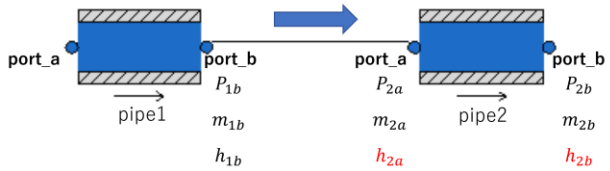
pipe2.port_a.h_{2a}, h_{2b}についてのエネルギー保存則

$$m_{2a} \times \mathrm{actualStream}(h_{2a}) + m_{2b} \times \mathrm{actualStream}(h_{2b}) = 0$$
$$h_{2a} = h_{2b}$$

actualStreamオペレータの定義式

$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

actualStreamオペレータの実装例 – 簡単な例



h_{2a}, h_{2b}の計算

pipe2.port_aに流体が流入する場合($m_{2a} > 0$)、エンタルピーの保存式よりh_{2a}, h_{2b}は次式で計算できます。

$$m_{2a} \times \mathrm{actualStream}(h_{2a}) + m_{2b} \times \mathrm{actualStream}(h_{2b}) = 0$$
$$h_{2a} = h_{2b}$$

actualStreamオペレータの定義式を適用

$$m_{2a} \times \frac{\max(-m_{1b}, \epsilon) h_{1b}}{\max(-m_{1b}, \epsilon)} + m_{2b} \times h_{2b} = 0$$
$$h_{2a} = h_{2b}$$

h_{2a},h_{2b}について整理

$$h_{2b} = -\frac{1}{m_{2b}} \left(m_{2a} \times \frac{\max(-m_{1b}, \epsilon) h_{1b}}{\max(-m_{1b}, \epsilon)} \right)$$
$$h_{2a} = h_{2b}$$

質量保存則より

$m_{1b} + m_{2a} = 0$
$m_{2a} + m_{2b} = 0$

$$h_{2b} = h_{1b}$$
$$h_{2a} = h_{2b}$$

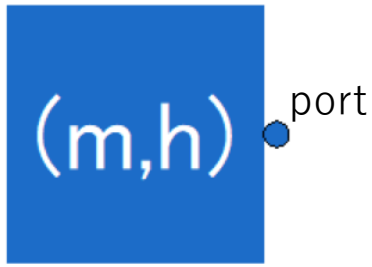
actualStreamオペレータの定義式

$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

actualStreamオペレータを使用してh_{2a}, h_{2b}がエンタルピーの保存式から適切に導出されていることが分かります。

境界(ソース)モデルの実装例

境界条件となるソースモデルにはポートへ直接値を代入するためinStream, actualStreamオペレータを使用せずに実装できます。



質量流量境界モデル

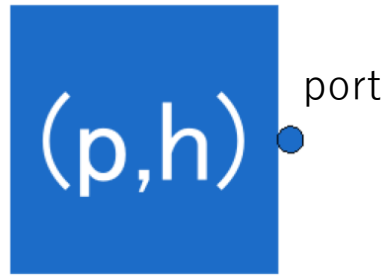
パラメータ*

・質量流量 m_flow

・比エンタルピー h

実装コード

```
port.m_flow = -m_flow;
port.h_outflow = h;
```



圧力境界モデル

パラメータ*

・圧力 p

・比エンタルピー h

実装コード

```
port.m_flow = -m_flow;
port.h_outflow = h;
```

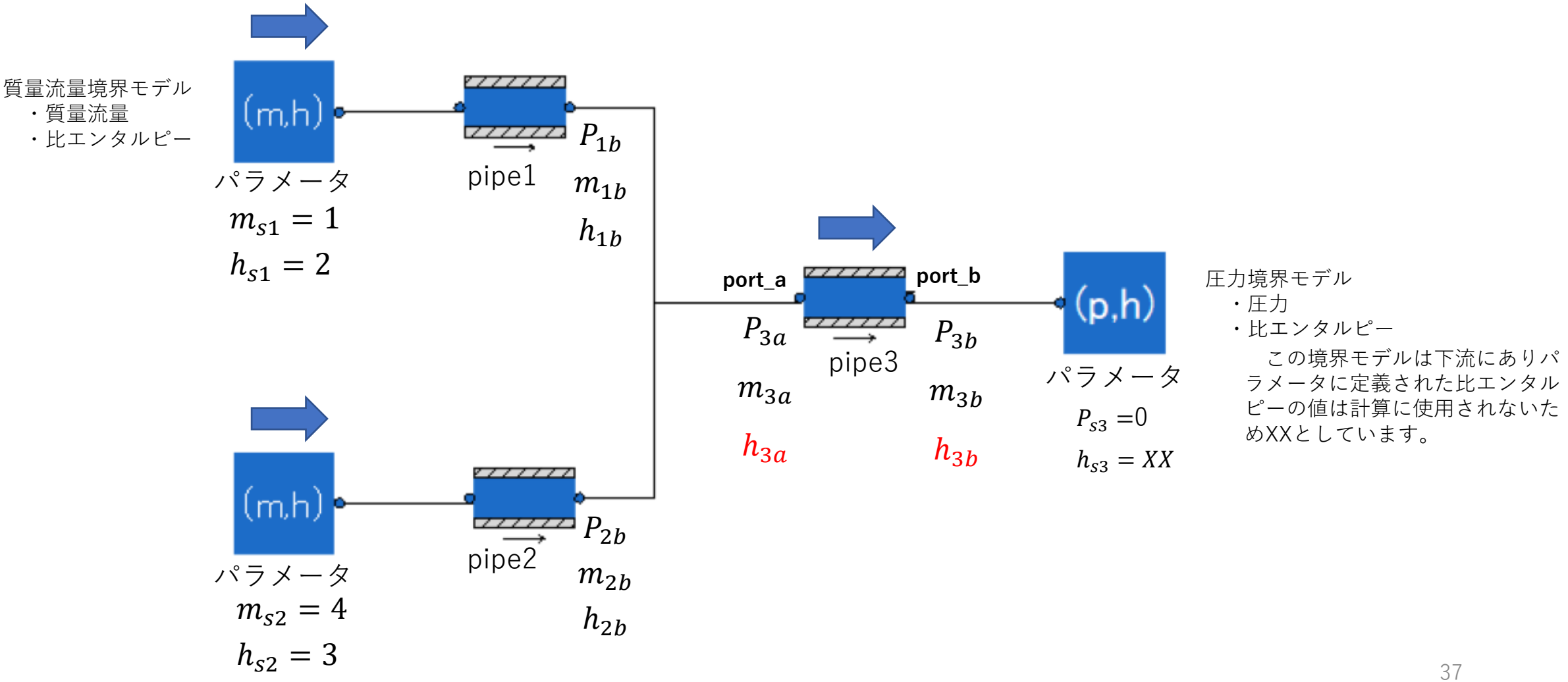
各境界モデルが上流となる場合、ポートの比エンタルピーの値は接続されたモデルに受け渡されますが下流となる場合は受け渡されないため無視しても構いません。

* Modelicaコードではinputで宣言されていますが、分かりやすさを優先して「パラメータ」と記載しています。

actualStreamオペレータの実装例 – 合流がある場合

Exercise1

以下のpipe3の比エンタルピーh_{3a}, h_{3b}を手計算で求めてください。

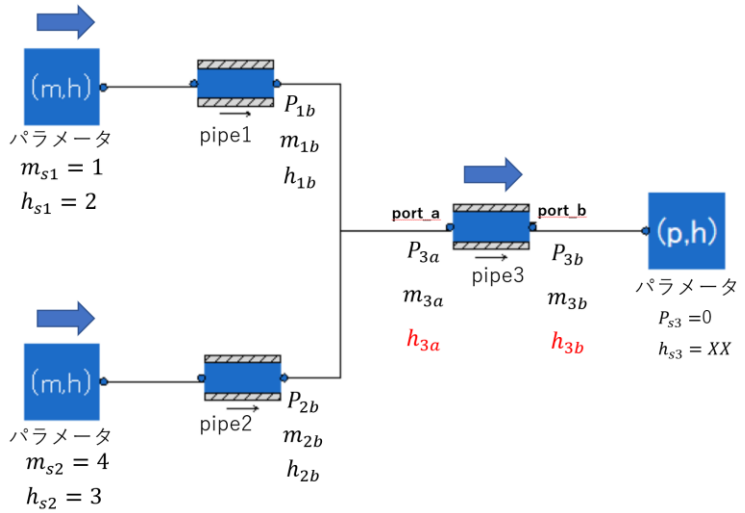


actualStreamオペレータの実装例 – 合流がある場合

解答1

Pipe3のエネルギー保存則

$$m_{3a} \times \mathrm{actualStream}(h_{3a}) + m_{3b} \times \mathrm{actualStream}(h_{3b}) = 0$$
$$h_{3a} = h_{3b}$$



actualStreamオペレータの定義式を適用

$$m_{3a} \times \left(\frac{\max(-m_{1b}, \epsilon)\, h_{1b} + \max(-m_{2b}, \epsilon) h_{2b}}{\max(-m_{1b}, \epsilon) + \max(-m_{2b}, \epsilon)} \right) + m_{3b} \times h_{3b} = 0$$
$$h_{2a} = h_{2b}$$

h_{3a},h_{3b}について整理

$$h_{3b} = \frac{-m_{1b} h_{1b} - m_{2b} h_{2b}}{m_{3b}}$$
$$h_{3a} = h_{3b}$$

値を代入

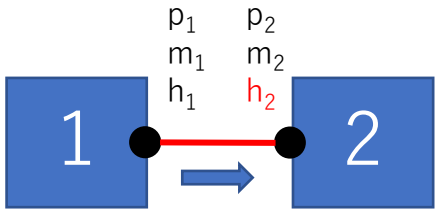
$$h_{3b} = \frac{1 \times 2 + 4 \times 3}{1 + 4} = 2.8$$
$$h_{3a} = 2.8$$

actualStreamオペレータを使用してh_{3a}, h_{3b}がエネルギー保存則から適切に導出されていることが分かります。

目次

inStreamオペレータの実装例

本資料では、左図のフローチャートに従ってstream変数の解説を行います



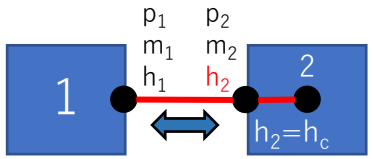
分岐・合流無し　一方向流れ

stream変数が表す物理現象

$$h_2 = \frac{-m_1 h_1}{-m_1}$$

ゼロ点の回避

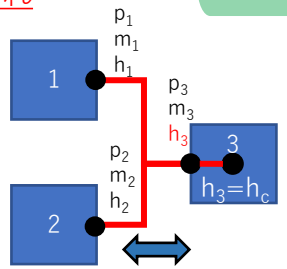
$$h_2 = \frac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)}$$



分岐・合流無し　双方向流れ

双方向流れ(逆流)への対応

$$h_2 = \begin{cases} \dfrac{\max(-m_1,\epsilon) h_1}{\max(-m_1,\epsilon)} & (m_2 > 0) \quad \text{流入} \\ h_c & (m_2 < 0) \quad \text{流出} \end{cases}$$



分岐・合流有り　双方向流れ

分岐・合流

$$h_3 = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_3 > 0, \quad i = 1,2) \quad \text{流入} \\ h_c & (m_3 < 0) \quad \text{流出} \end{cases}$$

Modelica Language

コーディング

実装時のルールとオペレータ

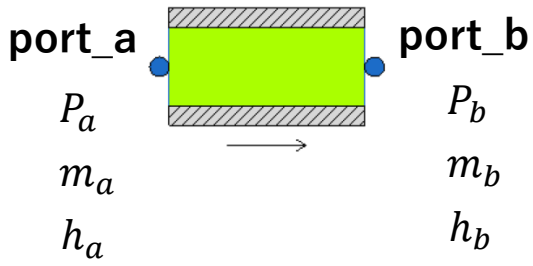
$$actualStream(h_i) = \begin{cases} \dfrac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} & (m_i > 0, \quad j \neq i) \\ h_i & (m_i \leq 0) \end{cases}$$

or

$$inStream(h_i) = \frac{\sum_j \max(-m_j,\epsilon)\, h_j}{\sum_j \max(-m_j,\epsilon)} \qquad (j \neq i)$$

inStreamオペレータの実装例 – サンプルモデルの作成

inStreamオペレータの使用方法を確認しましょう。
inStreamオペレータを使ったSimplePipeクラスの比エンタルピーは以下のように実装できます。



SimplePipe

実装式

質量保存則

$$m_a + m_b = 0$$

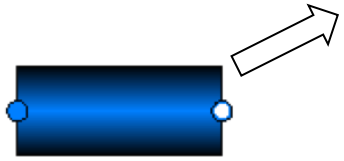
圧力-質量流量の関係式

$$P_a - P_b = Km_a^2$$ （パラメータ K : pressure drop coeficient）

エンタルピーの保存式に対応する計算式

$$h_a = inStream(h_b)$$
$$h_b = inStream(h_a)$$

$$inStream(h_i) = \frac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} \quad (j \neq i)$$



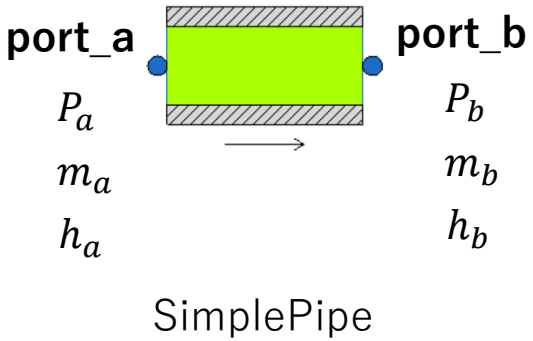
Modelica.Fluid.Pipes.StaticPipeにも近い形の計算式が使用されています

port_aに接続されたコネクタから計算される比エンタルピーはport_bに代入しport_bは上記と逆となっています。
このように実装することで条件分岐が無くなり計算が簡便になります。
しかしこの実装にはいくつかの注意事項があります。

inStreamオペレータの実装例 – サンプルモデルの作成

Exercise2

StreamConnectors.Components.Pipeクラスのエンタルピーの保存式に関する式を
以下の実装式のように修正してSimplePipeクラスを作成してください。
またアイコンの色を変更してください。



SimplePipe

実装式

エネルギー保存則に対応する計算式

$$h_a = inStream(h_b)$$
$$h_b = inStream(h_a)$$

$$inStream(h_i) = \frac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} \quad (j \neq i)$$

inStreamオペレータの実装例 – サンプルモデルの作成

解答2

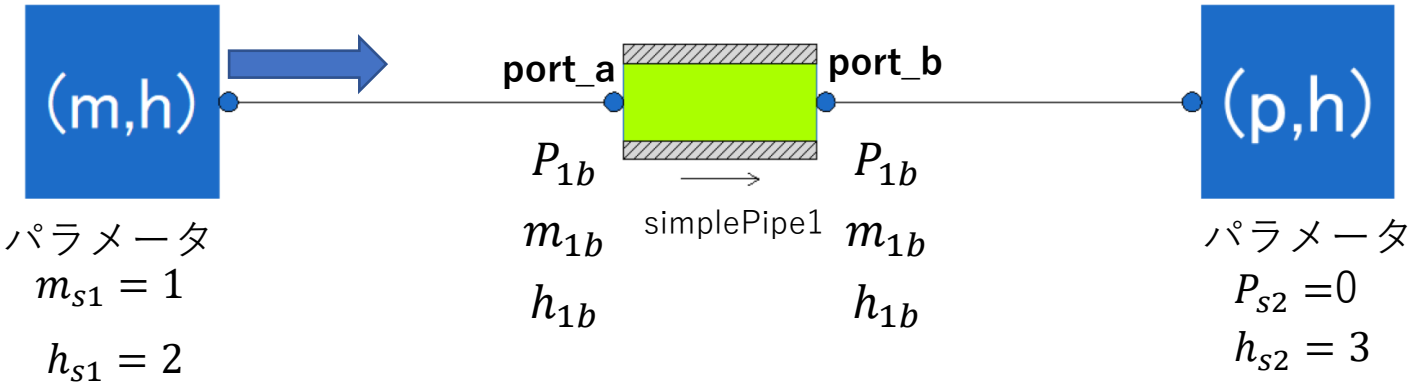
```
model SimplePipe "A simple static pipe with heating/cooling"
  input Real Q_flow=0 "Heat flow rate into the pipe [kW]" annotation(Dialog(group="Heat input"));
  parameter Real K=dp_nominal/m_flow_nominal^2 "Pressure drop coefficient";
  parameter Real dp_nominal=0.5 "Nominal pressure drop [bar]"
    annotation (Dialog(group="Nominal values"));
  parameter Real m_flow_nominal=1 "Nominal mass flow rate [kg/s]"
    annotation (Dialog(group="Nominal values"));

  StreamConnectors.Interfaces.FluidPort port_a annotation (Placement(transformation(extent={{-120, ...);
  StreamConnectors.Interfaces.FluidPort port_b annotation (Placement(transformation(extent={{100, ...);
equation
  // Mass balance
  port_a.m_flow + port_b.m_flow = 0;

  // Momentum balance
  port_a.p - port_b.p = K*port_a.m_flow*abs(port_a.m_flow);

  // Energy balance
  port_a.h_outflow=inStream(port_b.h_outflow);
  port_b.h_outflow=inStream(port_a.h_outflow);

  annotation ( ...);
end SimplePipe;
```

エネルギー保存則に対応する計算式

アイコンの色については省略

inStreamオペレータの実装例 – 簡単な例

inStreamオペレータを使用した場合の計算順序を確認しましょう。



圧力境界モデル
- 圧力
- 比エンタルピー

入力した比エンタルピーはポートに与えられます。今回の場合、下流側の輸送量が計算に使用されます。

h_{1a}, h_{1b}の計算式

$$h_{1a} = inStream(h_{1b})$$
$$h_{1b} = inStream(h_{1a})$$

inStreamオペレータの定義式

$$inStream(h_i) = \frac{\sum_j \max(-m_j, \epsilon)\, h_j}{\sum_j \max(-m_j, \epsilon)} \quad (j \neq i)$$

inStreamオペレータの定義式から考えるとh_{1a}, h_{1b}に入る値はいくらになるでしょうか？

inStreamオペレータの実装例 – 簡単な例

h_{2a}, h_{2b}の計算

$$h_{1a} = inStream(h_{1b})$$
$$h_{1b} = inStream(h_{1a})$$

actualStreamオペレータの定義式を適用

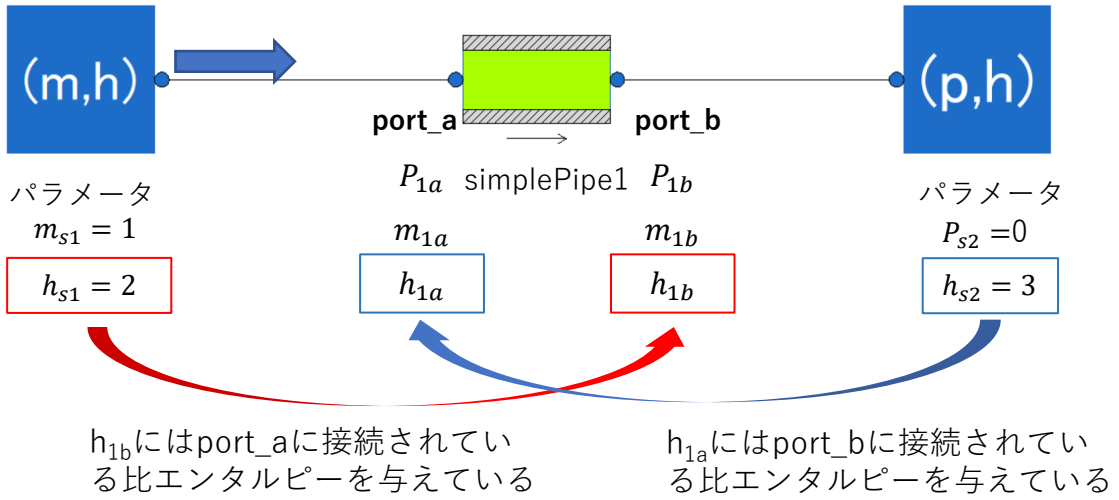
$$h_{1a} = \frac{\max(-m_{s2}, \epsilon) h_{s2}}{\max(-m_{s2}, \epsilon)}$$
$$h_{1b} = \frac{\max(-m_{1b}, \epsilon) h_{s1}}{\max(-m_{1b}, \epsilon)}$$

h_{1a},h_{1b}について整理

$$h_{1a} = h_{s2}$$
$$h_{1b} = h_{s1}$$

値を代入

$$h_{1a} = 3$$
$$h_{1b} = 2$$

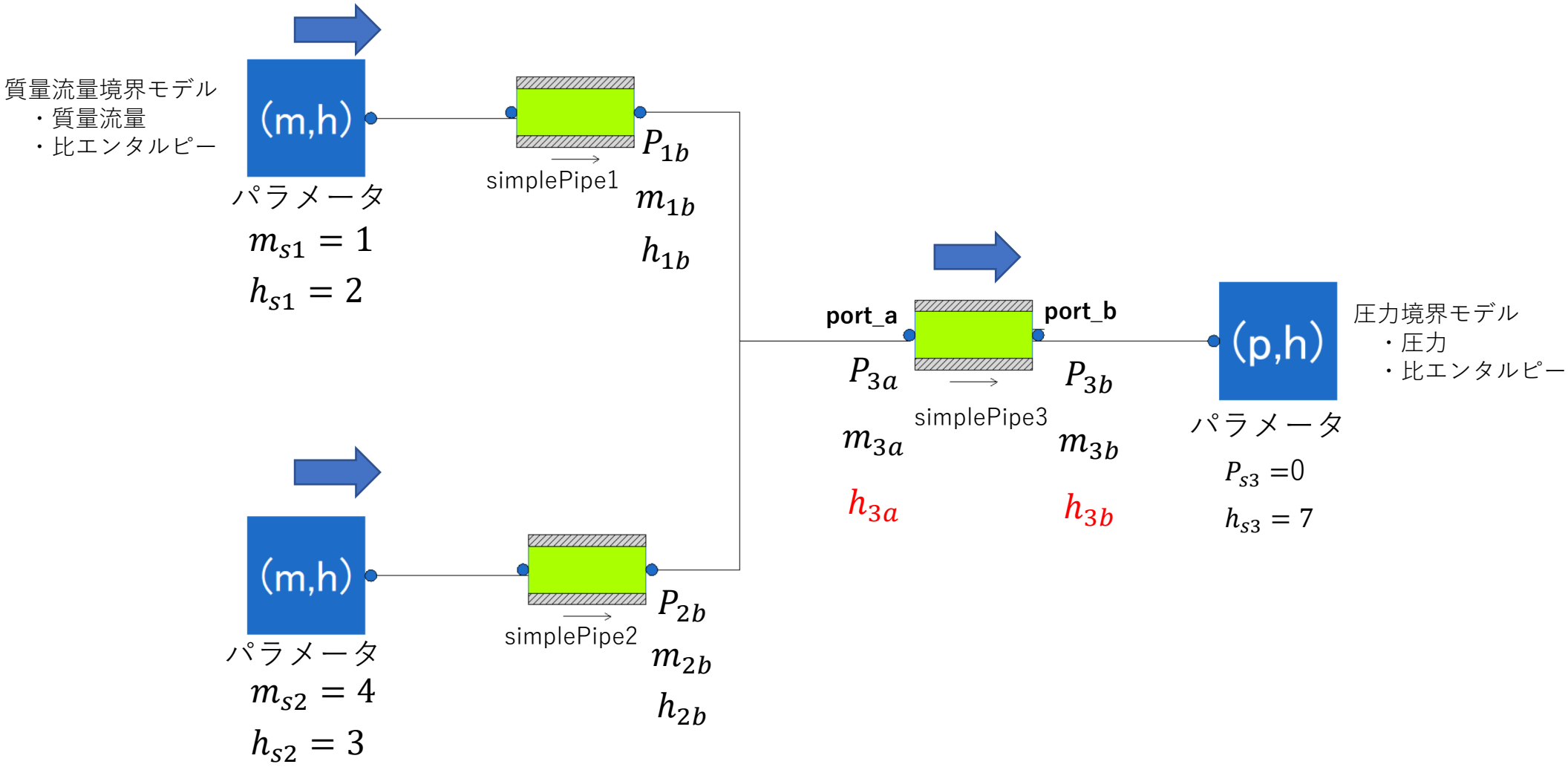


simplePipe1モデルから流出する比エンタルピーh_{1b}は適切に計算されていますが
流入する比エンタルピーh_{1a}は不自然な計算式となっています。
上記は一例ですが他の場合も同様にstream変数の計算結果を確認する際は注意が必要となります。

inStreamオペレータの実装例 – 合流がある場合

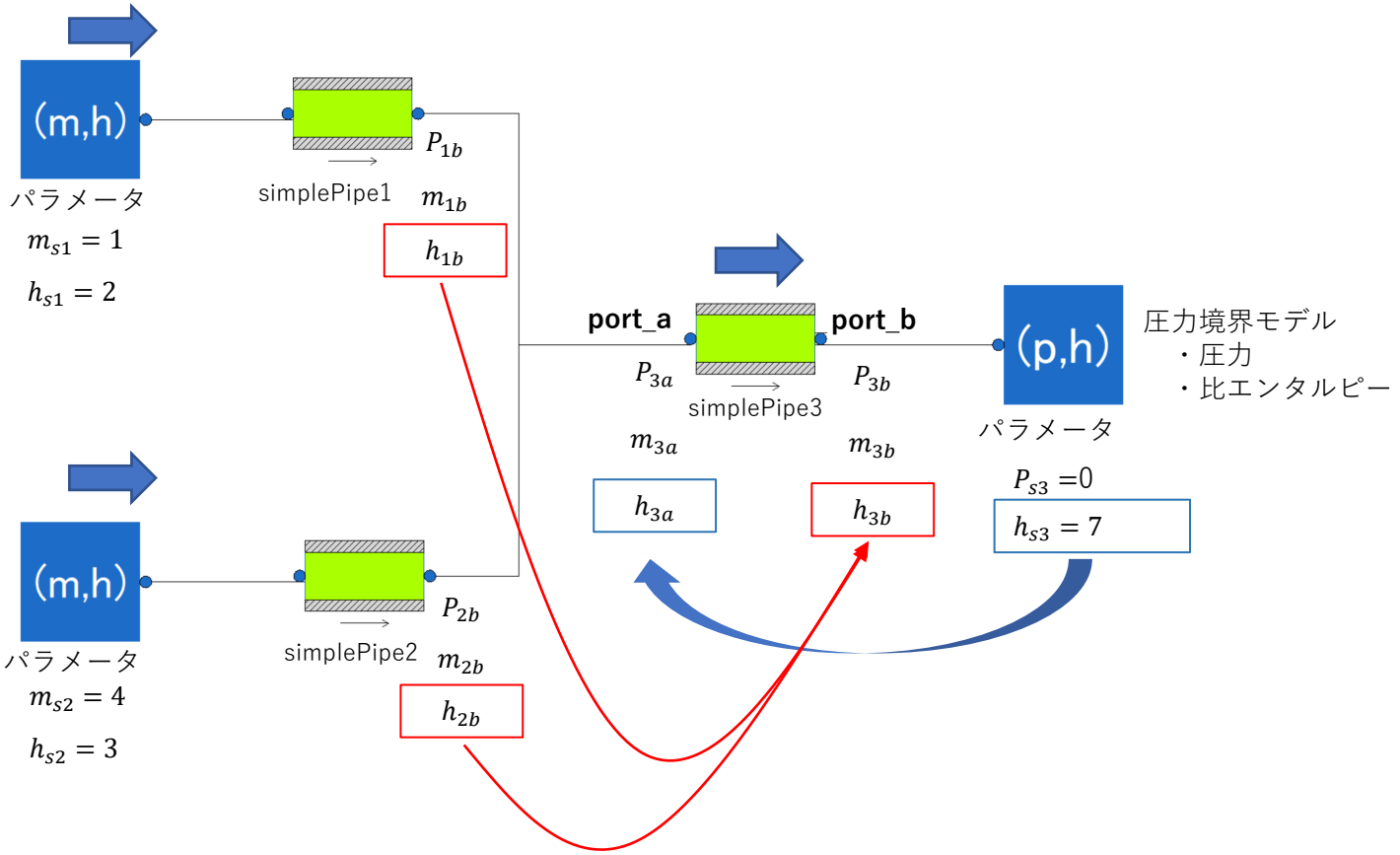
Exercise3

以下のsimplePipe3の比エンタルピーh$_{3a}$, h$_{3b}$を手計算で求めてください。



inStreamオペレータの実装例 – 合流がある場合

解答3



$$h_{3a} = inStream(h_{3b})$$
$$h_{3b} = inStream(h_{3a})$$

途中式は省略

$$h_{3a} = h_{s3}$$
$$h_{3b} = \frac{m_{1b}h_{1b} + m_{2b}h_{2b}}{m_{1b} + m_{2b}}$$

値を代入

$$h_{3a} = 7$$
$$h_{3b} = \frac{14}{5}$$

inStreamオペレータを使用すると下流の値が上流のポートに入力されることに気をつけてください。

コラム：比エンタルピーの変数名がh_outflowである意味

StreamConnectorsライブラリやMSL.Fluidライブラリのポート内で宣言されている比エンタルピーの変数名はh_outflow(比エンタルピーの流出量)となっています。
これはstream変数は流出量を計算することが目的であることを表しています。

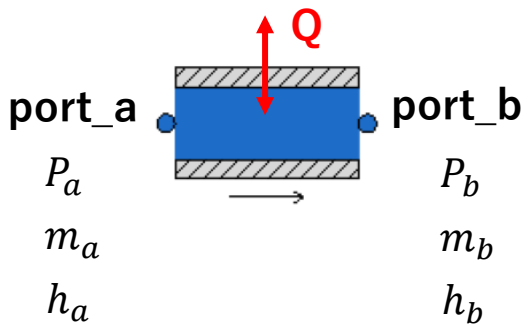
actualStream、inStreamオペレータを使用した場合流入するstream変数の値が適切に計算されない場合があります。
そのようなことを考慮して、変数名にoutflowをつけて流出量を計算することが目的であることを示しているのかと思います。


```
connector FluidPort "Simple stream connector.“
  Real p "Potential/effort variable";
  flow Real m_flow "Flow variable";
  stream Real h_outflow "Specific enthalpy";
end FluidPort;
```

目次

熱の流入出がある場合

熱の流入出がある場合を検討してみましょう。Pipeモデルは熱流量Qの流入出が考慮できるようになっています。Qはパラメータとして宣言しています。



StreamConnectors.Components.Pipe

可読性のため以下のように置き換えています。

P_X：port_X.p
m_X：port_X.m_flow
h_X：port_X.h_outflow
X：ポート名

実装式

説明のため式は一部簡易化し、エンタルピーの保存式のみを表示しています

エンタルピーの保存式

流入出エンタルピーの保存

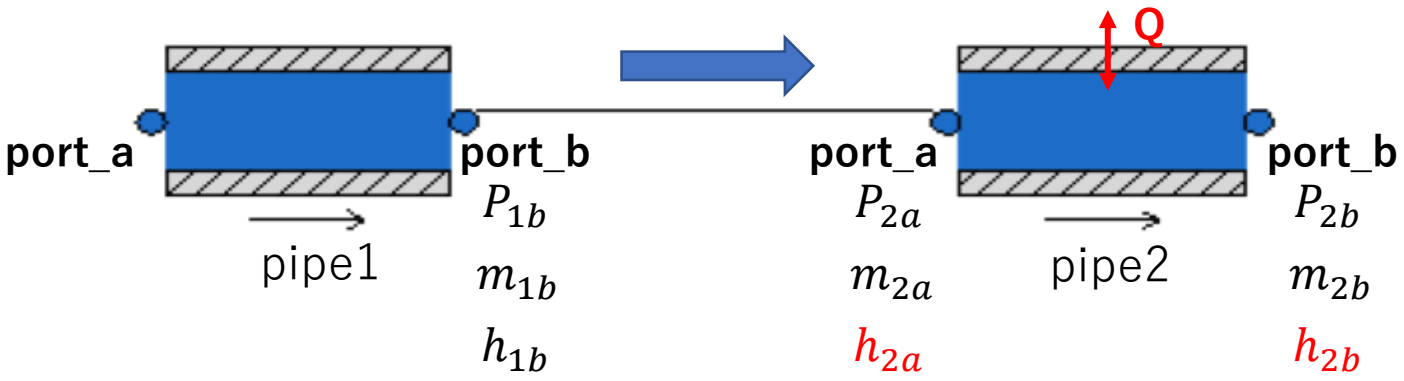
$$m_a \times \mathrm{actualStream}(h_a) + m_b \times \mathrm{actualStream}(h_b) + Q = 0$$

上記のままだとh_aかh_bの値が未定となるためダミー値を入力する

$$h_a = h_b$$

熱の流入出がある場合 – 簡単な例

下図のような場合にh_{2a}, h_{2b}に着目してactualStreamオペレータの計算順序を確認しましょう。



赤字：未知変数
黒字：既知変数

$$m_{2a} \times \text{actualStream}(h_{2a}) + m_{2b} \times \text{actualStream}(h_{2b}) + Q = 0$$
$$h_{2a} = h_{2b}$$

途中式は省略

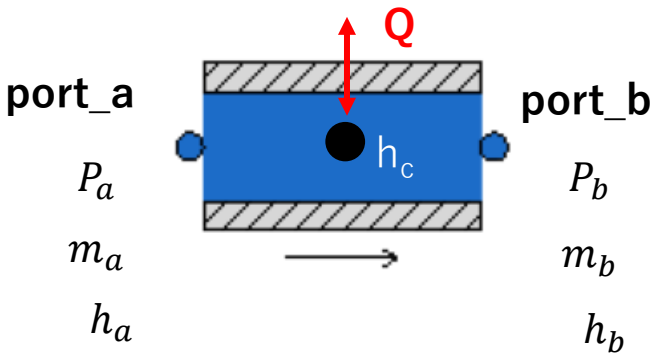
$$h_{2b} = h_{1b} + \frac{Q}{m_{2a}}$$
$$h_{2a} = h_{2b}$$

正しい答えは
$\boldsymbol{h_{2a} = h_{1b}}$

上記のように、h_{2b}は適切に計算されているがh_{2a}は不自然な計算式となっている。しかし、stream変数は流出量を計算することが目的であることから使用上の問題はないことが分かる。

熱の流入出がある場合 – 検討案

これまで見てきた方法ではポートのstream変数に不自然な結果が返ってきてしまい不便です。
以下のように実装することで自然な結果が返ってくるようになります。




```
//h_cはパイプの中央部分の比エンタルピー
h_c = semiLinear(sign(m_a), inStream(h_a), -inStream(h_b))+Q/abs(m_a);
h_a = semiLinear(sign(m_a), inStream(h_a), h_c);
h_b = semiLinear(sign(m_b), inStream(h_b), -h_c);
```

ポートのstream変数に自然な結果が返ってくるためには他に適切な方法があると思います。
色々と検討してみてください。

参考資料

(1)　Francesco Casella, Martin Otter, Michael Sielemann and Rüdiger Franke, Stream Connectors - An Extension of Modelica for Device-Oriented Modeling of Convective Transport Phenomena, 7th Modelica Conference(2009)

(2)　Modelica Specification 3.4

(3)　Overview and Rationale for Modelica Stream Connectors, https://doc.modelica.org/Modelica%204.0.0/Resources/Documentation/Fluid/Stream-Connectors-Overview-Rationale.pdf (参照日 2021/8/22)