Robust Simulation of Stream-Dominated Thermo-Fluid Systems: From Directed to Non-Directed Flows

Dirk Zimmer*, Niels Weber, Michael Meißner

DLR German Aerospace Center, Institute of System Dynamics and Control, Münchener Strasse 20, 82234 Oberpfaffenhofen, Germany; *dirk.zimmer@dlr.de

SNE 31(4), 2021, 177-184, DOI: 10.11128/sne.31.tn.10581
Received: 2021-11-15 (selected EUROSIM 2019 Postconf. Pub., extended version); Accepted: 2021-11-20
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna, ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. A new robust and efficient formulation for stream-dominated thermal fluid systems has been developed and published as open-source library. This methodology has been predominantly designed for and applied on systems with known flow direction. Since it is not directly evident how it is transferred to systems with unknown flow direction, this paper details the implementation in the corresponding library. As an example, a reversible heat-pump is presented.

Introduction

What are stream dominated systems? Let us suppose we have a component where a fluid is flowing from a set of inlets to a set of outlets. A pipe section with one inlet and one outlet is a simple representation of such a component. When the mass flow rate is high, the fluid within the component will be quickly replaced by the fluid stream of the inlet(s).

In such a case, it is often a reasonable idealization to assume that the thermodynamic state of the outlet Θ_{out} is algebraically coupled to the thermodynamic state of the inlet Θ_{in} (it may also depend on mass flowrate $\dot{m}$ and internal states $\mathbf{x}$ and inputs $\mathbf{u}$). If so, we denote the formulation of equations for such a component as stream-dominated. Correspondingly, a system or subsystem may be denoted as stream-dominated when it is (primarily) composed out of such components.

$$\Theta_{out} = f(\Theta_{in}, \dot{m}, \mathbf{x}, \mathbf{u}) \tag{1}$$

The reason is evident: with an algebraic coupling, any change in the thermodynamic state of the inlet has an immediate effect to the state of the outlet. This is never true for an actual physical system but as an idealization it may be upheld if the stream of the fluid is dominating over the capacity.

Stream dominance is a very useful idealization and hence frequently used. The corresponding algebraic equations enable to describe even complex thermodynamic processes very efficiently using very few time-dependent state variables or even none at all.

Because of this efficiency, various tools (often denoted as 1D tools) support the object-oriented modeling and simulation of thermal fluid systems. Examples are the optimal control of power plants, the simulation of building physics and the environmental control systems (ECS) for cars or aircraft; the very last one being the authors' application domain. Figure 1 shows a picture of a three wheel bootstrap cycle, a classic construction as part of the environmental control system of many civil aircraft. The example has been created using a proprietary library written in the open object-oriented language Modelica [1].

The authors recently published the DLR ThermofluidStream Library [2], an open-source implementation of the stream-dominated approach. This library contains a special sub-package that implements the concepts for non-directed flows described in this paper. The reader is invited to study the code of this implementation as additional content to this paper.

1 On Stream-dominated Systems

Although, stream-dominance may lead to a purely algebraic system that can be efficiently solved in theory, it is often difficult or inefficient to solve in practice. The system of Figure 1 is a good example.

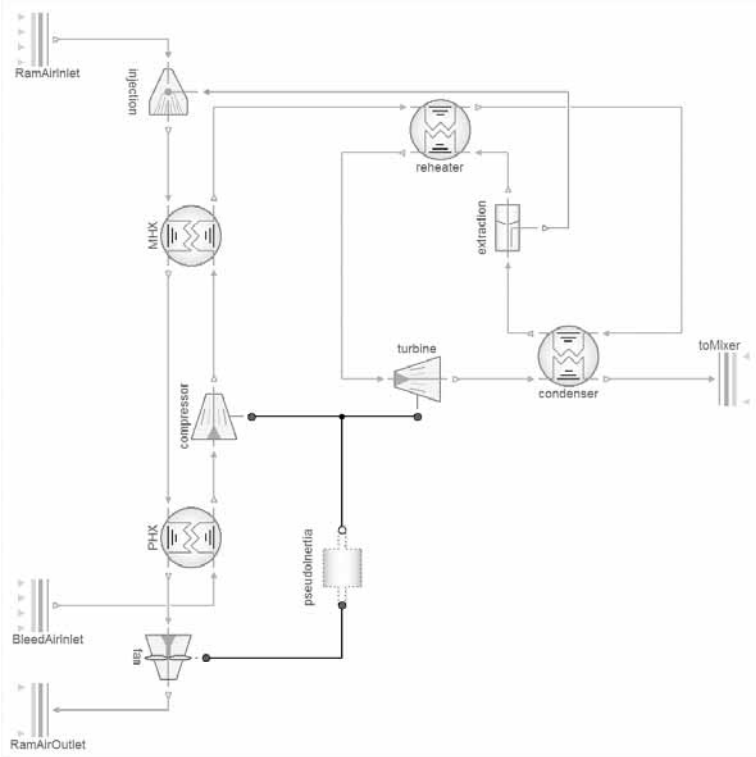

Figure 1: Modelica diagram of a three-wheel-bootstrap cycle. The hot and dense bleed-air is cooled against the outside ram air. The energy of the expansion is used to increase cooling efficiency. Furthermore the air is dehumidified. The system is built extremely compact and the stream of air dominates. The direction of the stream is known a priori.

When modeled purely by algebraic equations using the object-oriented language Modelica, a non-linear system of more than 200 equations results that needs to be solved iteratively by a numerical method [3]. Simulation tools such as Dymola [4] may automatically reduce the dimension to 40 but yet alone finding the area of convergence remains a serious problem.

The high-degree of non-linearity hence poses a serious robustness problem for the object-oriented modeling of stream-dominated systems. Attempts to solve this by more advanced numerical solvers (such as homothopy methods [5]) had been so far of limited success.

Fortunately, a recent advance led to a more robust formulation of stream-dominated systems. The idea is outlined and tested in [6] and further implemented and elaborated in [7] and [8]. Here, we quickly repeat the core idea which centers around the decomposition of the pressure p into the inertial pressure r and the steady-mass flow pressure $\hat{p}$

$$p = \hat{p} + r \tag{2}$$

For a mass-flow $\dot{m}$ that is constant along the stream direction ds, the difference in inertial pressure Δr is purely defined by the geometry of the flow and independent of the thermodynamic state:

$$-\Delta r = \frac{d\dot{m}}{dt} \int \frac{ds}{A_\mathrm{s}} \tag{3}$$

where A_s represents the flow cross section at position s. Hence changes in mass flow rate $\dot{m}$ can be determined by a linear system of equations, once the gradients in steady mass flow pressure $\hat{p}$ between individual streams and at the boundaries are known. The steady-mass flow pressure $\hat{p}$ is an unusual term, not present in text-books on the matter. It is simply defined as the complement of r to p. For a steady mass flow (meaning $d\dot{m}/dt = 0$), $\hat{p} = p$ and hence its name. Fortunately, for many applications, it is feasible to express the thermodynamic state using $\hat{p}$

When doing so, the equations for a stream-dominated system can be set up in a very favorable form. The highly non-linear computations of the thermodynamic state can be arranged in an explicit order

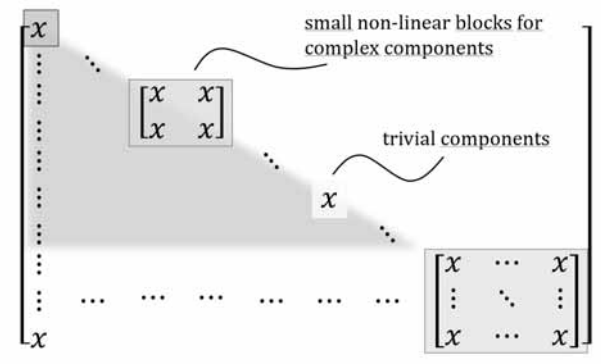

Figure 2: The structure of the resulting equation system. The blue part forms a non-linear LU system that can be computed downstream (barring deliberate small exceptions on component level). It computes all $\hat{p}$ and the thermodynamic state. The green system is linear and computes r and $d\dot{m}/dt$.

going downstream from sources to sinks. The changes in the individual mass flow rates are then computed solving a system of linear equations. This scheme is illustrated by the corresponding BLT-form in Figure 2. Again see [6] and [7] for more details.

When upholding these rules and realizing this scheme in an object-oriented modeling framework such as Modelica, it is reflected in the design of the component interface (denoted as connector in Modelica). The thermodynamic state is represented as a signal going in direction of the stream from source to sink. The inertial pressure r and the mass flow rate $\dot{m}$ form a pair of effort and flow. Here is the corresponding code of an inlet and an outlet in Modelica:

```
connector Inlet
 replaceable package Medium = Modelica.
     Media.Interfaces.PartialMedium;
 SI.Pressure r;
 flow SI.MassFlowRate m_flow;
 input Medium.ThermodynamicState
     state_hat;
end Inlet;

connector Outlet
 replaceable package Medium = Modelica.
     Media.Interfaces.PartialMedium;
 SI.Pressure r;
 flow SI.MassFlowRate m_flow;
 output Medium.ThermodynamicState
     state_hat;
end Outlet;
```

The definition of the thermodynamic state state_hat can differ for different media but commonly consists of pressure $\hat{p}$, specific enthalpy $\hat{h}$ as well as mass fractions for media with more then one component. In any case the state is expressed by quantities for the steady mass flow as described above.

This scheme has been applied with great success for the modeling and simulation of thermofluid systems with directed flow such as modern aircraft ECS. Using this scheme, robustness of the models could be drastically improved [6] since no large non-linear equation system needs to be solved iteratively anymore. With respect to performance, the approach is also interesting. Especially for real-time simulation of such systems, first investigations reveal promising results [7].

In order to be concise, we cannot review the complete built-up of the equation system here (please be referred to [6] and [7]) but for our analysis in Section 2, we need to focus on two items:

- How the linear equations for r and $\dot{m}$ are interlinked with the non-linear computation for $\hat{p}$.
- The structural prerequisite to ensure the LU-form of the non-linear part.

On the first point: the non-linear computation of $\hat{p}$ leads to differences in pressure that are compensated by the inertial pressure r in the following way:

- For each inlet boundary of the stream: $r = 0$ and hence $\hat{p} = p_{\text{inlet}}$ (4)
- For each outlet boundary of the stream: $\hat{p} + r = p_{\text{outlet}}$ (5)
- For each split of a mass flow $\dot{m}_0$ into $\dot{m}_1 \ldots \dot{m}_\text{n}$:
  $\hat{p}_1 = \hat{p}_2 = \ldots = \hat{p}_\text{n} = \hat{p}_0$ (6)
  and
  $r_1 = r_2 = \ldots = r_\text{n} = r_0$ (7)
- For each junction of mass flows $\dot{m}_1 \ldots \dot{m}_\text{n}$ into $\dot{m}_0$:
  $\hat{p}_0 = g_{\text{mix}}(\hat{p}_1, \ldots, \hat{p}_\text{n})$ (8)
  and
  $\hat{p}_1 + r_1 = \hat{p}_2 + r_2 = \ldots = \hat{p}_\text{n} + r_\text{n} = \hat{p}_0 + r_0$ (9)

Where g_{mix} represents the weighted average of the steady mass flow pressures $\hat{p}_1, \ldots, \hat{p}_\text{n}$ where the corresponding volume flow rates V_i form the weigths. To be well-natured, the function g_{mix} shall also be regularized against zero and negative mass (and volume) flow rates. Here is one possible implementation using a small ε for regularization:

$$g_{\mathrm{mix}} = \frac{\sum_{\mathrm{i}} (|V_{\mathrm{i}}| + \varepsilon) \hat{p}_{\mathrm{i}}}{\sum_{i} (|V_{\mathrm{i}}| + \varepsilon)} \tag{10}$$

To ensure the LU-form of the non-linear part, the directed stream must form an acyclic graph. If there are cycles (such as in a vapor cycle) then the cycle must be torn apart by volume elements. The inlet of a volume element then acts as an outlet boundary of the stream and the outlet of the volume element acts as inlet boundary of the stream. This works because the internal state of a volume prevents a direct algebraic equation between volume inlet and outlet (at the expense of time-dependent states).

2 Non-directed Fluid Streams

So far, this approach has only been applied to 1D fluid streams of known flow direction; hence directed flow. For many aircraft system, such an approach suffices completely.

Some other systems however, like the aircraft bleed(-air) system can also be modeled as 1D system but the flow direction is a priori unknown. Bleed air may flow from the engine to the central hub during normal operation but in the other direction for engine start-up. Such systems hence have non-directed flows. This paper presents an extension of the above scheme.

When extending stream-dominated systems from directed to non-directed flows, one inherently is confronted with a fundamental problem: the underlying assumption of stream dominance will inevitably be violated.

Any transient from a strong positive mass flow to a strong negative mass flow passes through zero mass flow. No matter how one quantitatively defines stream dominance, at zero mass flow, this assumption cannot be upheld anymore and the algebraic coupling between "outlets" and "inlets" loses all of its validity. And indeed, one shall not apply a stream-dominated modeling approach if the splish-splash behavior of a fluid at low mass-flow rates is of any interest.

However, for many applications, it is of no interest and the transient just leads from one stream dominated operation point to another stream dominated operation point. Hence, validity at zero-mass flow is not needed. It suffices when the model is robust and well-natured so that the transient does not break or corrupt the simulation. This is the first challenge.

The second challenge is of structural nature. In a system of directed flow, the fluid stream is represented by a signal flow for the thermodynamic state (see the inputs/outputs in the Modelica connector). Since the direction of the fluid flow is given, also the direction of the signal flow is predetermined.

A change in fluid flow direction hence also implies a change in signal flow direction. The algebraic equation systems must hence be structured in such a way that it supports both flow directions. A well proven solution for this is to double the signal flow as it is also applied for the Modelica Standard Fluid library [9, 10, 11].

Following this approach, each two port element (such as a pipe) has a signal flow in both directions. Depending on the actual mass flow rate, one of these signals is chosen as relevant whereas the other signal represents a dummy signal.

Consequently, the interface of a component for non-directed flows has now two signals: one for the thermodynamic state when the flow direction is out of the component and one for the thermodynamic state when the flow direction is into the component. There still remains the pair of effort and flow, formed by the inertial pressure and mass flow rate.

```
connector FluidPort
 replaceable package Medium =  Modelica.
      Media.Interfaces.PartialMedium;
 SI.Pressure r;
 flow SI.MassFlowRate m_flow;
 inpput Medium.ThermodynamicState
     state_hat_in;
 output Medium.ThermodynamicState
     state_hat_out;
end Inlet;
```

2.1 Cycle-free graphs for the signal flow of the thermodynamic state

In graph-theoretical terms, we thus convert a non-directed graph into a directed graph so that each non-directed edge is being replaced by two directed edges in opposite directions. However, we have to do this cleverly since as stated in section 2, the signal flow must be loop free in order to ensure that no non-linear equation system occurs and that the lower-triangular form can be maintained. This means that a loop-free non-directed graph must remain loop-free after conversion.

In order to understand the problem, let us first study what not to do. Figure 3 shows a straight forward, naïve conversion from an undirected graph to a directed graph: a cycle of two edges is being created between junctions A and B.

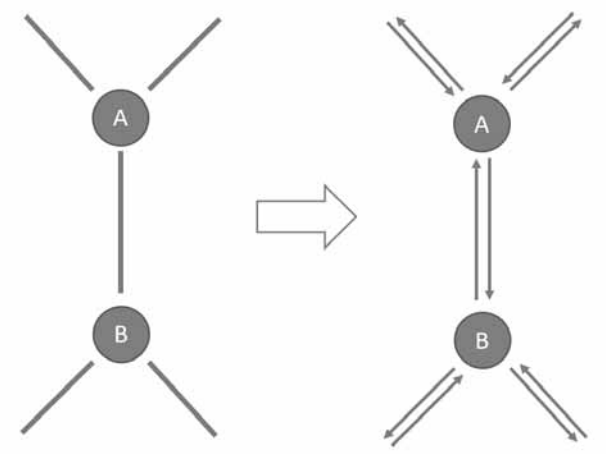

Figure 3: Naïve conversion from an undirected graph to a directed graph for expressing potential signal flows for the thermodynamic state.

The cycle in Figure 3 is however an artefact of conversion and not a loop of the actual system of fluid streams. A flow cannot flow from B to A while it flows from A to B. We have to incorporate this structural knowledge into the conversion and we do so by splitting up the junctions A and B and creating a separate virtual junction for each potential outflow. In this way, we can explicitly state that any outflow must be structurally independent from the flow in its directly opposing direction. Figure 4 depicts how to do this:

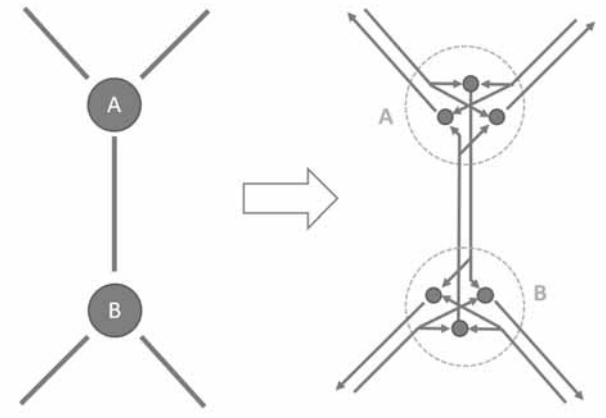

Figure 4: This conversion avoids the creation of cycles by going from a cycle-free undirected graph to a cycle-free directed graph but still expresses all potential information flows for the thermodynamic state. It however contains more nodes and edges.

The computation of the thermodynamic state based on $\hat{p}_{\text{in}}$ (hypothetical) downstream direction hence follows the structure of such a directed graph. Evidently this computation is always performed for both directions although only one direction can actually be relevant. Hence care must be taken that formulas used for the downstream direction are robust against mass flows in opposite direction (they do not need to be valid but should be well-natured).

2.2 Regularization scheme for the inertial pressure at boundaries

The linear equations for the inertial pressure r need to be reformulated as well. As described in Section 1, the boundary equations (4) and (5) for r differed from an inlet to an outlet. For a non-directed system, it is not predetermined what is an outlet and what is an inlet. Hence, the equation has to be unified for a general boundary and made dependent on the flow direction expressed by the sign of the mass flow rate $\dot{m}$. A straight forward implementation would be:

$$r = \text{if } \dot{m} > 0 \text{ then } p_{\text{Boundary}} - \hat{p} \text{ else } 0 \qquad (11)$$

However, such a hard switch is not feasible since the partial derivative $\partial r/\partial \dot{m}$ shall be bounded in order to enable numerical stability of explicit ODE solvers and a sufficient area of convergence for implicit ODE solvers. Hence a regularization scheme needs to be applied that expresses a continuous transition between the two flow directions. ε is used to express the size of this transition region in terms of mass flow rate. We then use the regstep-function

$$y = \text{regstep}(x, y1, y2, \varepsilon) \qquad (12)$$

as depicted in Figure 5 to softly interpolate between y_1 for positive x, and y_2 for negative.

Given this function we can reformulate the boundary equation for r in regularized form:

$$r = \text{regstep}(\dot{m}, p_{\text{Boundary}} - \hat{p}, 0, \varepsilon) \qquad (13)$$

Please note that there is no physical basis for the applied regularization. This means that when the simulation computes in the zone of regularization $(-\varepsilon < \dot{m} < \varepsilon)$ the validity of the model may be (at least partially) lost. As stated before, this is a remedy solution for short term transients going through zero mass flow.

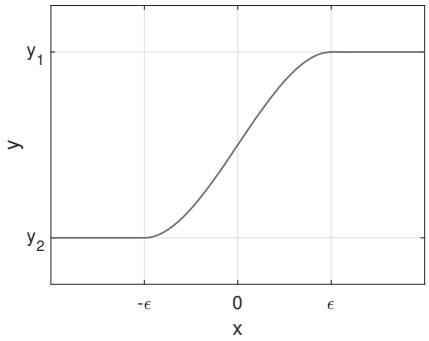

Figure 5: regstep-function used to smoothly interpolate between two values y_1 and y_2, depending on the sign of x.

2.3 Regularization scheme for the inertial pressure at junctions

A corresponding regularization scheme is also needed for the inertial pressures at the junctions. As in Section 1, the main point is to uphold the equivalence of pressure $p = \hat{p} + r$ for all flows at a junction. Because the flow direction is a priori unknown, there is no outflow indicated by the index 0 anymore (as in equation 9) and all flows (whether inflowing or outflowing) are indexed from 1 to n.

$$\hat{p}_1 + r_1 = \hat{p}_2 + r_2 = \ldots = \hat{p}_\mathrm{n} + r_\mathrm{n} \tag{14}$$

Due to the doubling of the signal flow, there is now a pair of steady mass flow pressure $(\hat{p}_{(\mathrm{i,in})}, \hat{p}_{(\mathrm{i,out})})$ for each inertial pressure r_i. The representative steady mass-flow pressure $\hat{p}_\mathrm{i}$ must hence be chosen according to the actual direction of the corresponding mass flow $\dot{m}_\mathrm{i}$. For the same reasons as in Section 2.2, this shall be done in a regularized form:

$$\hat{p}_\mathrm{i} = \mathrm{regstep}(\dot{m}, \hat{p}_{(\mathrm{i,in})}, \hat{p}_{(\mathrm{i,out})}, \varepsilon) \tag{15}$$

With this regularization in place, the linear equations for the inertial pressure can now be formulated without a priori knowledge of the flow direction. Please note that the non-linearity involving the regstep-function is irrelevant because the mass flow rate $\dot{m}$ always forms a state of the system and hence can assumed to be known. It is not part of the equation system, only its time derivative is.

3 Implementation in Modelica and Use Case Application

As we are now familiar with the theoretical background on how to set up the equations for non-directed thermo-fluid systems, we want to see how this can be applied in practice on an example system. Remembering that we are interested in stream-dominated systems in both flow directions, a reversible heat pump forms a suitable example system. In such heat pumps, the flow direction through the heat exchangers can be reversed and the validity at the zero-transition from positive to negative mass-flow is not of main interest. Reversible heat pumps are used for example in the thermal management of modern electric cars or for residential air conditioning. The latter will serve as our use case application.

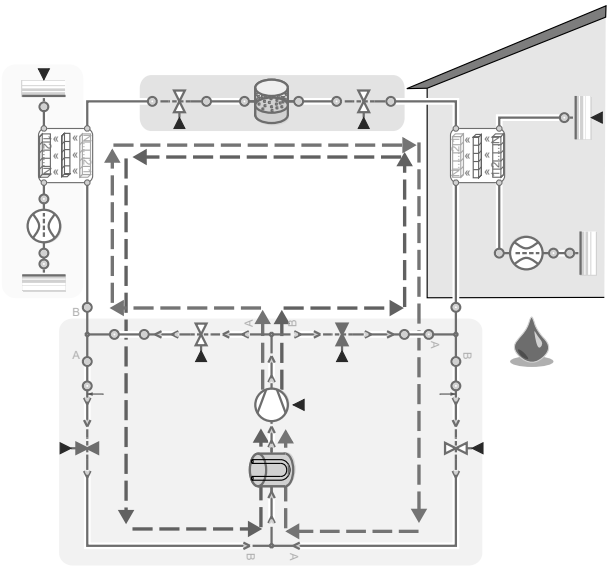

Figure 6: Example of a reversible heatpump that contains two non-directed heat exchangers.

Figure 6 gives an overview of the system architecture. The system consists solely of components from the freely available DLR ThermoFluidStream Library, which was recently published by the authors [2]. The main components of the system are known from a standard vapor cycle, as it can be found in every refrigerator. It consists of a compressor, condenser, expansion device and an evaporator. When the flow direction of the refrigerant is changed, the heat exchangers can act as evaporator or condenser according to the current flow direction. The system can be operated in two different cycle modes - heating (red arrows) or cooling (blue ar-

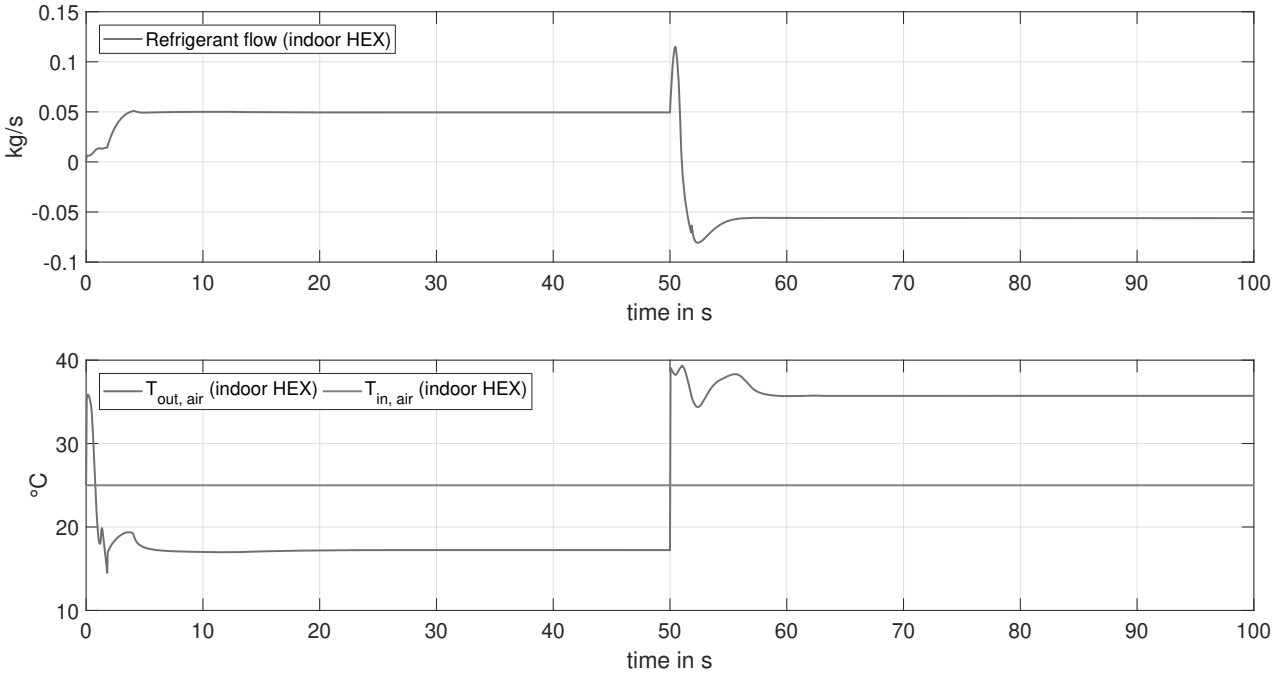

Figure 7: Simulation result showing the transient of flow reversal

rows). In cooling mode, the indoor unit acts as an evaporator and the outdoor unit acts as a condenser. Thus the heat is absorbed from the inside air and rejected to the outside. In heating mode, the cycle is reversed which makes the indoor unit the condenser and the outdoor unit the evaporator. Hence the heat is absorbed from the outside and rejected to the inside.

It is recommended to use components for non-directed flows only when the flow direction is really not known a priori. This is only the case for the heat exchangers, the phase-separator and the expansion devices (magenta). The two expansion valves are required to control the superheating temperature after the evaporator in both operating modes. In practice, the change of flow direction is carried out by a so-called reversing valve. In our simulation model, the flow direction is controlled by a system of valves and non-directed junctions (yellow). The flow direction through the compressor and the accumulator does not change during switching, which is also the case in real systems hence those components can be kept directed.

To conclude this section, let us have a look at an exemplary simulation result. For the sake of simplicity and to spare the model of an additional controller, we set the compressor speed to a fixed value. With the expansion valves, the superheating temperature is controlled to $5K$ and the temperature at the air side of the indoor unit is set to $25°C$.

Looking at the results in Figure 7 we can observe that we are able to switch from cooling to heating mode at $t = 50s$ during simulation quite drastically. The plot shows a sudden change of the temperature at one end of the heat exchanger. This highlights the stream dominated modeling approach where the replacement of the fluid is neglected on purpose and the model reacts hence quicker than a real device would do.

From a computational point of view though, the transition from positive to negative mass flow through the indoor unit does not cause any problems. Nevertheless the system needs some time to reach the steady state during startup and after switching. This can mainly be referred to the controller to reach the setpoint for the superheating temperature and the time constants for temperature adaption in the heat exchangers. After all, this example shows that the underlying methodology can robustly be applied for system architectures with undirected flow components.

4 Conclusions

In previous papers, a new scheme for the robust computation of directed thermofluid systems has been presented. It proved to be extremely useful for our modelling and simulation activities for aircraft systems. Development time for system models could be drastically reduced and hard real-time simulation of complete systems became feasible. Yet, it was unclear whether this computational scheme (and the way to set up the equations) can be conveyed to (or even combined with) non-directed use cases.

This paper demonstrates that this is possible. It is not trivial and requires to take into account the structure of information flow (for the thermodynamic state) at the junctions and to apply a regularization scheme for low mass-flows and situations of flow reversal.

Once implemented in the equations of Modelica components, the proposed solution becomes easy to apply for the end-user. The only major concern for the end-user is that cycles of fluid streams shall be torn apart by volume elements. The open-source library [2] provides a corresponding implementation in Modelica and also includes the presented example.

Independent from the concrete implementation, two words of warning seem appropriate.

The first warning is a reminder that although non-directed flows are supported, stream dominance is still required for validity. Flow reversal shall occur only briefly during transient and should not be of main interest. This warning however also applies to similar popular modeling approaches as the Modelica Standard Fluid library [9, 10] that also implemented similar regularization schemes and relies on similar assumptions although this is unfortunately not prominently mentioned (honi soit qui mal y pense).

The second warning addresses a common misconception: one may think that because components for non-directed systems are more general than unidirectional components, it would be smart only to work with such components. However, models for directed systems yield far fewer cycles than their non-directed counterparts for the same reasons random directed graphs have fewer cycles than random undirected graphs. For instance, a bypass is not a cycle as long as the flow direction is known. Hence, we recommend to apply non-directed components only when necessary and to combine them with directed components where appropriate. Knowing the flow direction a priori is a piece of information too valuable to be thrown away.

References

[1] www.modelica.org

[2] Zimmer, D and Meißner, M and Weber, N: The DLR Thermofluid Stream Library *url: github.com/DLR-SR/ThermofluidStream* (2021).

[3] Zimmer D, Using Artificial States in Modeling Dynamic Systems: Turning Malpractice into Good Practice. In: *Proceedings of the 5th International Workshop on Equation-Based Object-Oriented Languages and Tools (EOOLT)* , Nottingham, United Kingdom (2013).

[4] Brück, Dag, Elmqvist, Hilding, Olsson, Hans: Dymola for Multi-engineering Modeling and Simulation. In: *Proc. of the 2nd International Modelica Conference*, Oberpfaffenhofen, Germany (2002).

[5] Casella F and Sielemann M und Savoldelli L (2011), Steady-state initialization of object-oriented thermo-fluid models for homotopy methods. In: *Proceedings of 8th International Modelica Conference. 8th International Modelica Conference*, 20.-22. Mar 2011, Dresden.

[6] Zimmer D. (2020), Robust Object-Oriented Formulation of Directed Thermofluid Stream Networks. In: *Mathematical and Computer Modelling of Dynamic Systems*, Vol 26, Issue 3.

[7] Otter M, et al.: Thermodynamic Property and Fluid Modeling with Modern Programming Language Constructs. In: *Proceedings of 8th International Modelica Conference. 13th International Modelica Conference*, Mar 04-06, 2019, Regensburg, Germany (2019).

[8] Zimmer D: Towards Hard Real-Time Simulation of Complex Fluid Networks. In: *Proceedings of the 13th International Modelica Conference* March 4-6, 2019, Regensburg, Germany (2019).

[9] Casella F et al.: The Modelica Fluid and Media library for modeling of incompressible and compressible thermo-fluid pipe networks, *Proceedings of the 13th International Modelica Conference* March 4-6, 2019

[10] Franke, Rüdiger, et. al.: Standardization of Thermo-Fluid Modeling in Modelica.Fluid. *Proceedings 7th Modelica Conference*, Como, Italy, (2009).

[11] Franke, Rüdiger, et. al.: Stream Connectors – An Extension of Modelica for Device-Oriented Modeling of Convective Transport Phenomena. *Proceedings 7th Modelica Conference*, Como, Italy (2009).