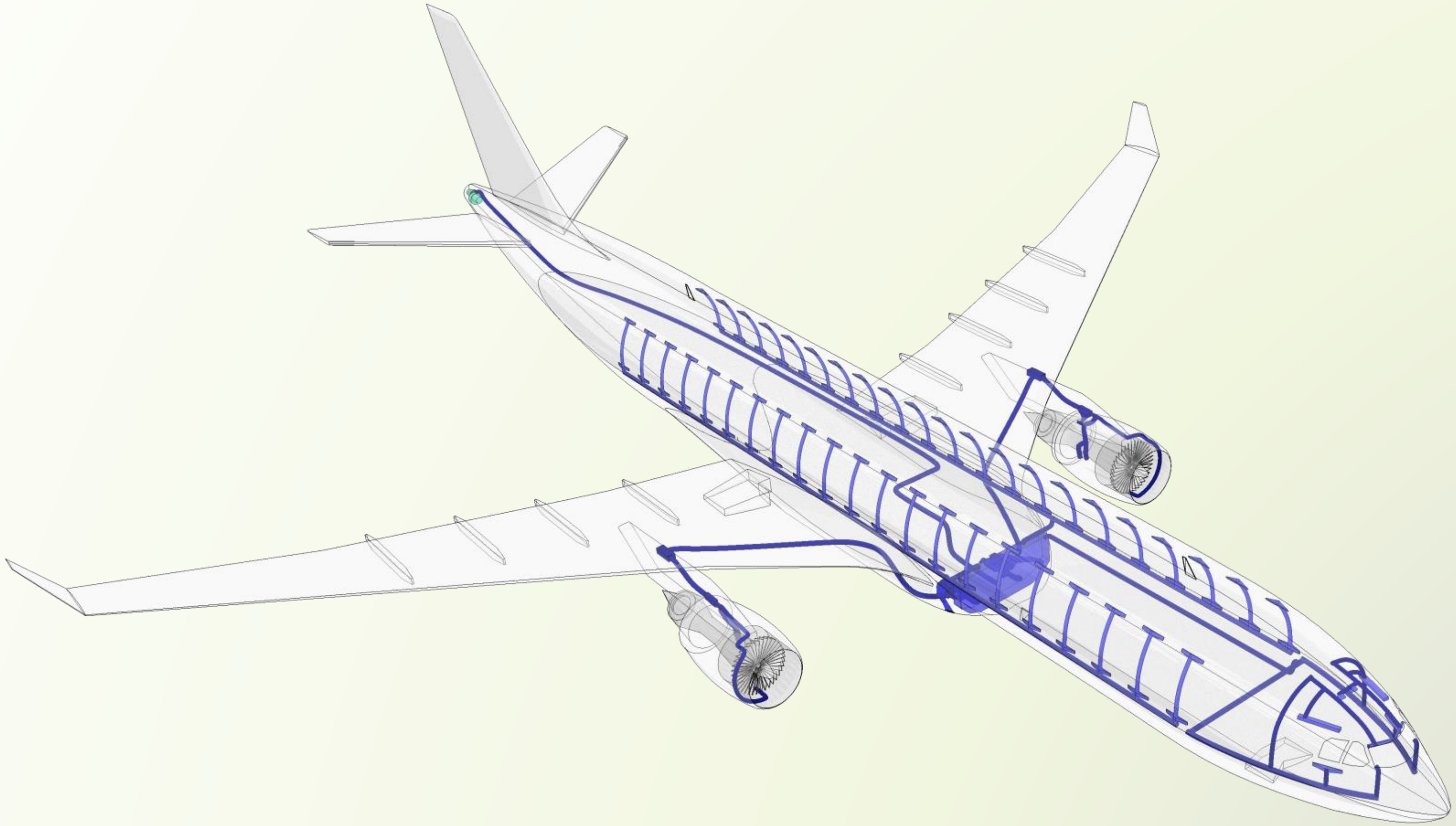
VOM HIMMEL HERAB GEFALLEN: WAS WIR ÜBER DIE ROBUSTE MODELLIERUNG AUS DER LUFTFAHRT LERNEN KÖNNEN

Dirk Zimmer, Institute of System Dynamics and Control, German Aerospace Center (DLR)
dirk.zimmer@dlr.de

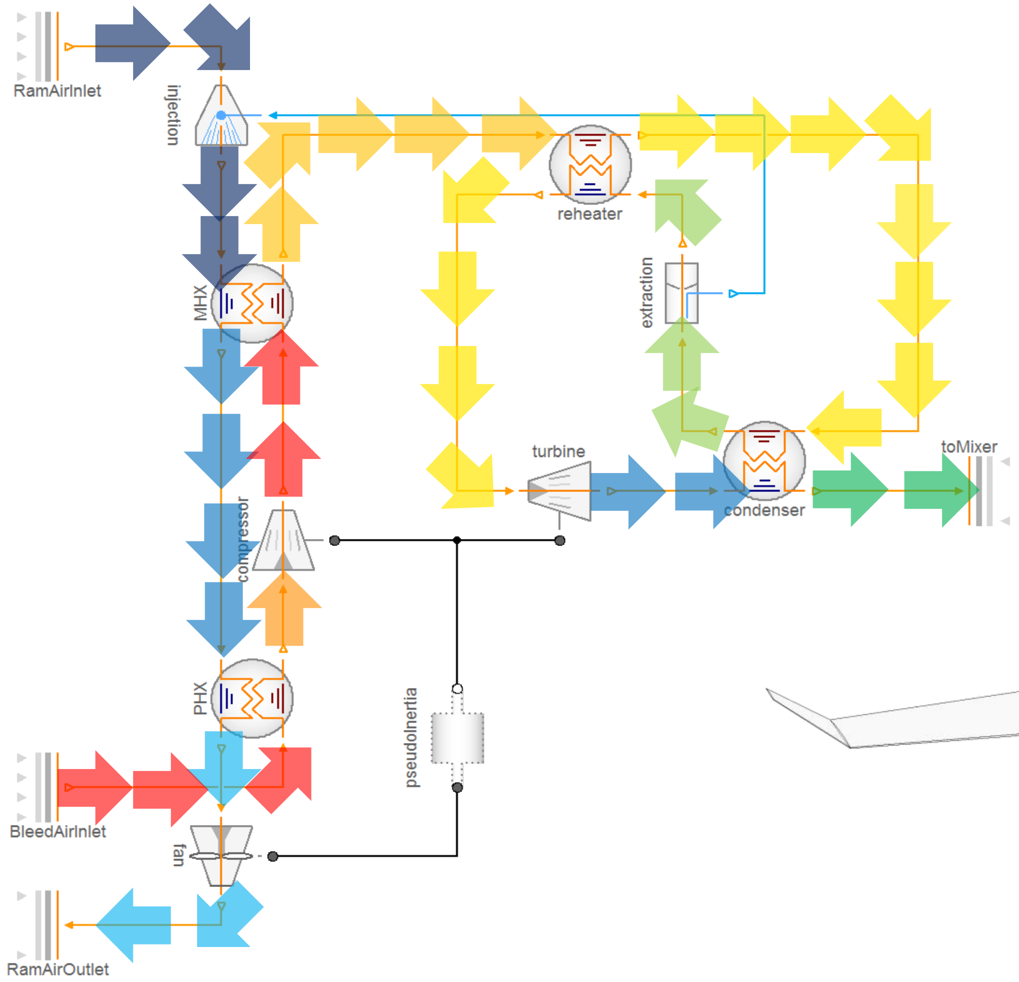
10.09.2024, ThermoSim 2024, München

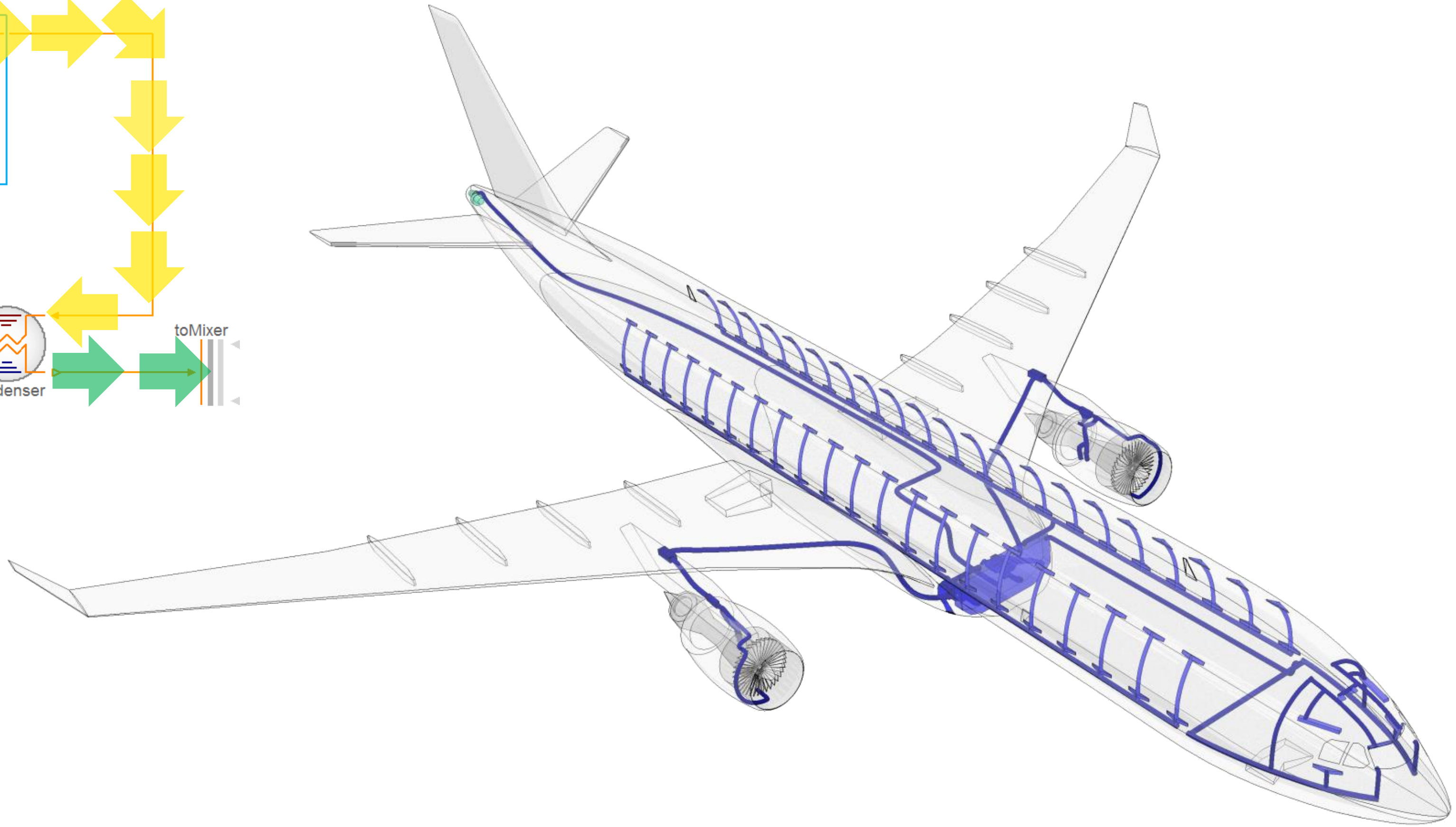
DLR



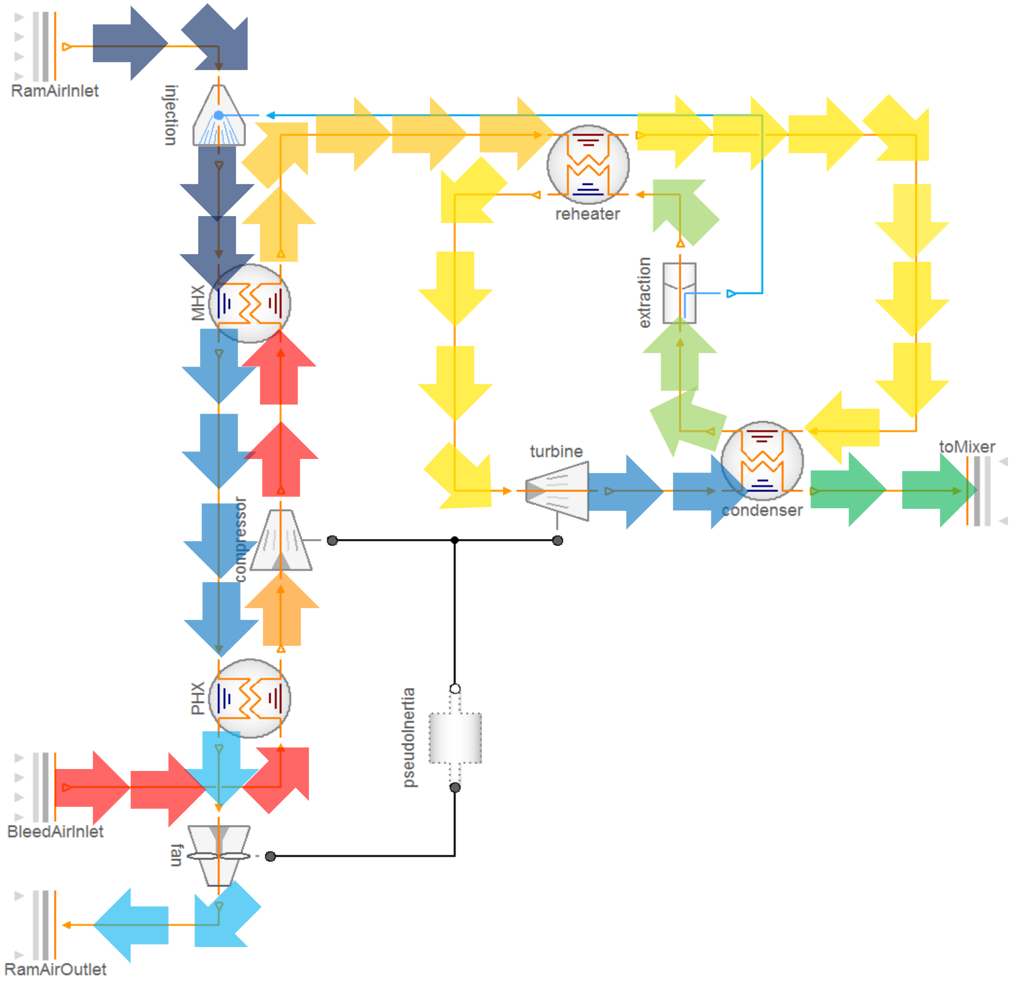
THERMOFLUID STREAMS

The Practical Problem





The Practical Problem



- Using text-book equations (as in standard libraries) leads to large non-linear equation systems.
- >200 variables or >40 dimensions for the numerical solver.
- Whether you attain a solution is completely uncertain and the computational effort may be substantial.
- Other applications in energy and buildings share this problem.

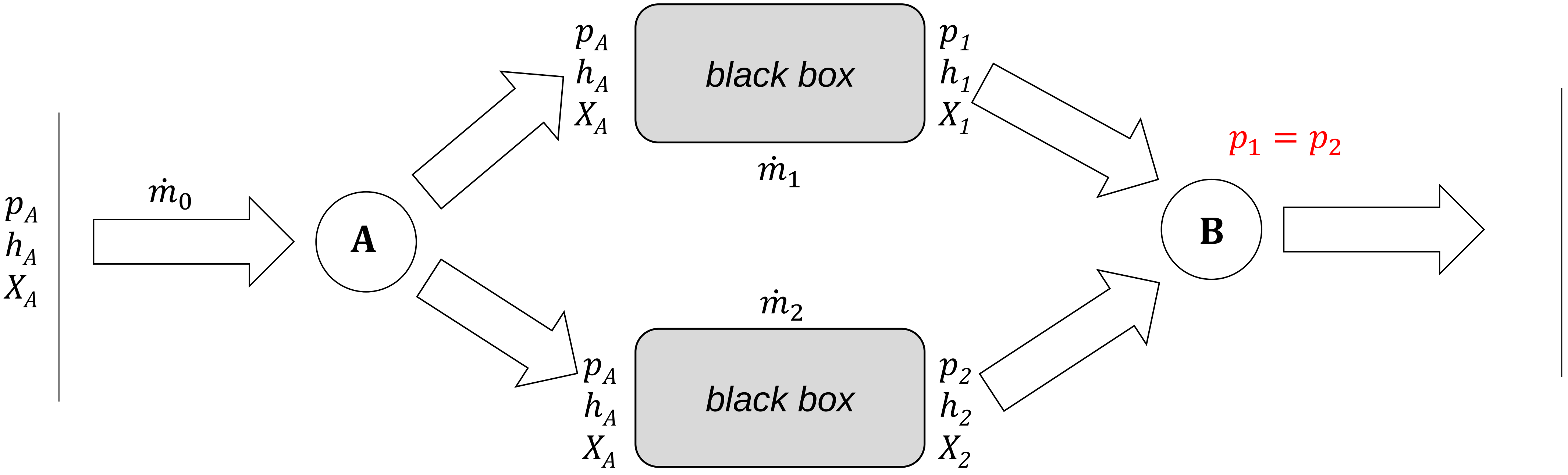
Simulation Error

A fatal exception ocurred at 027:C8127 by the non-linear equation system solver.
Here is a cryptic error code that is of absolutely no use: 420.
Simulation has been stopped to prevent damage from your virtual universe.

*press any key to acknowledge defeat
*press Ctrl+Alt+Del if you think that this is any better
*by the way, we deleted your hard-drive

Press any key to continue _

Understanding the Problem



- The constraint of pressure equivalence leads to a non-linear dependence on mass flow rates.

Understanding the Problem

„God does not play dice."

-Albert Einstein

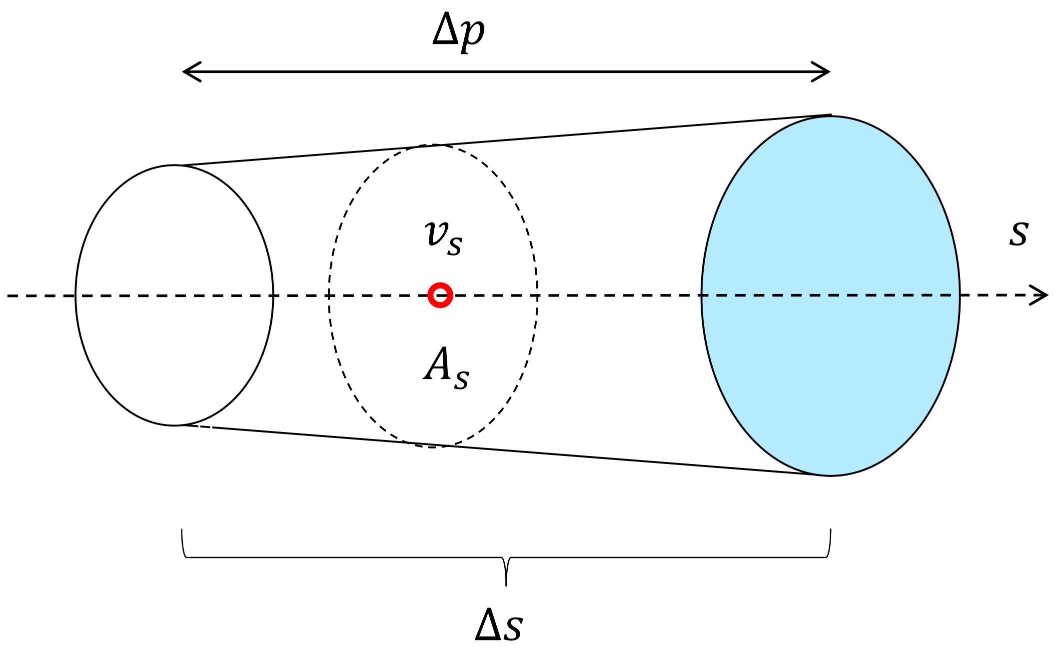
„God does not solve non-linear equation systems"

-Dirk Zimmer

We shall revisit the fundamental laws of physics and let them guide us to the solution.

The Inertial Pressure

$$p = \hat{p} + r \quad \text{with} \quad \Delta r = \frac{d\dot{m}}{dt} \int \frac{1}{A_s} ds$$

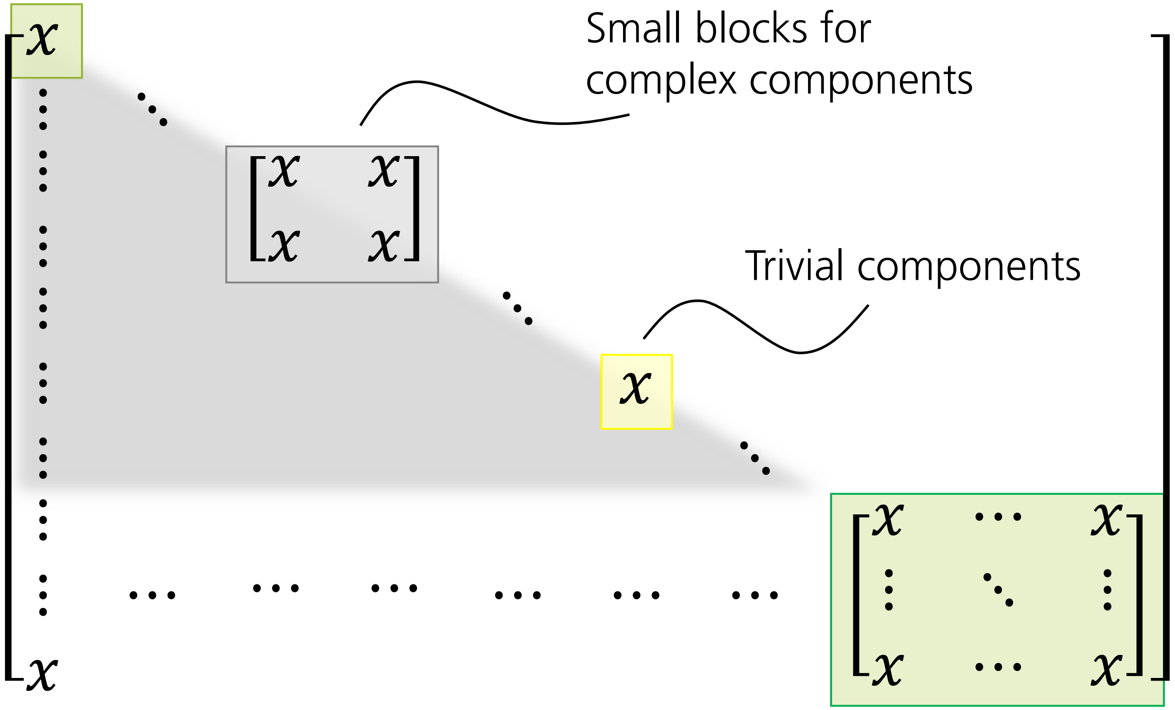


Structure of the resulting equation system

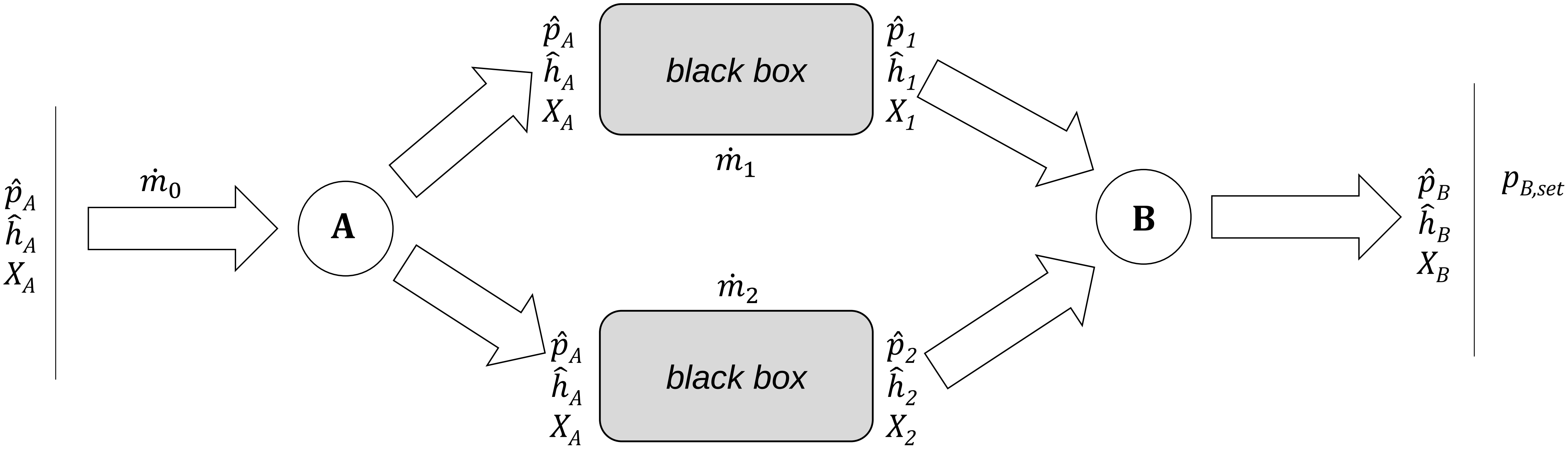
$$p = \hat{p} + r \qquad \text{with} \qquad \Delta r = \frac{d\dot{m}}{dt} \int \frac{1}{A_s} ds$$

- The inertial pressure is defined by a linear law that is independent of its thermodynamic state
  ➔ Only linear equations in implicit form

- When choosing a different spatial resolution for r then for $\hat{p}$, enables us to formulate all non-linear equations downstream in explicit form

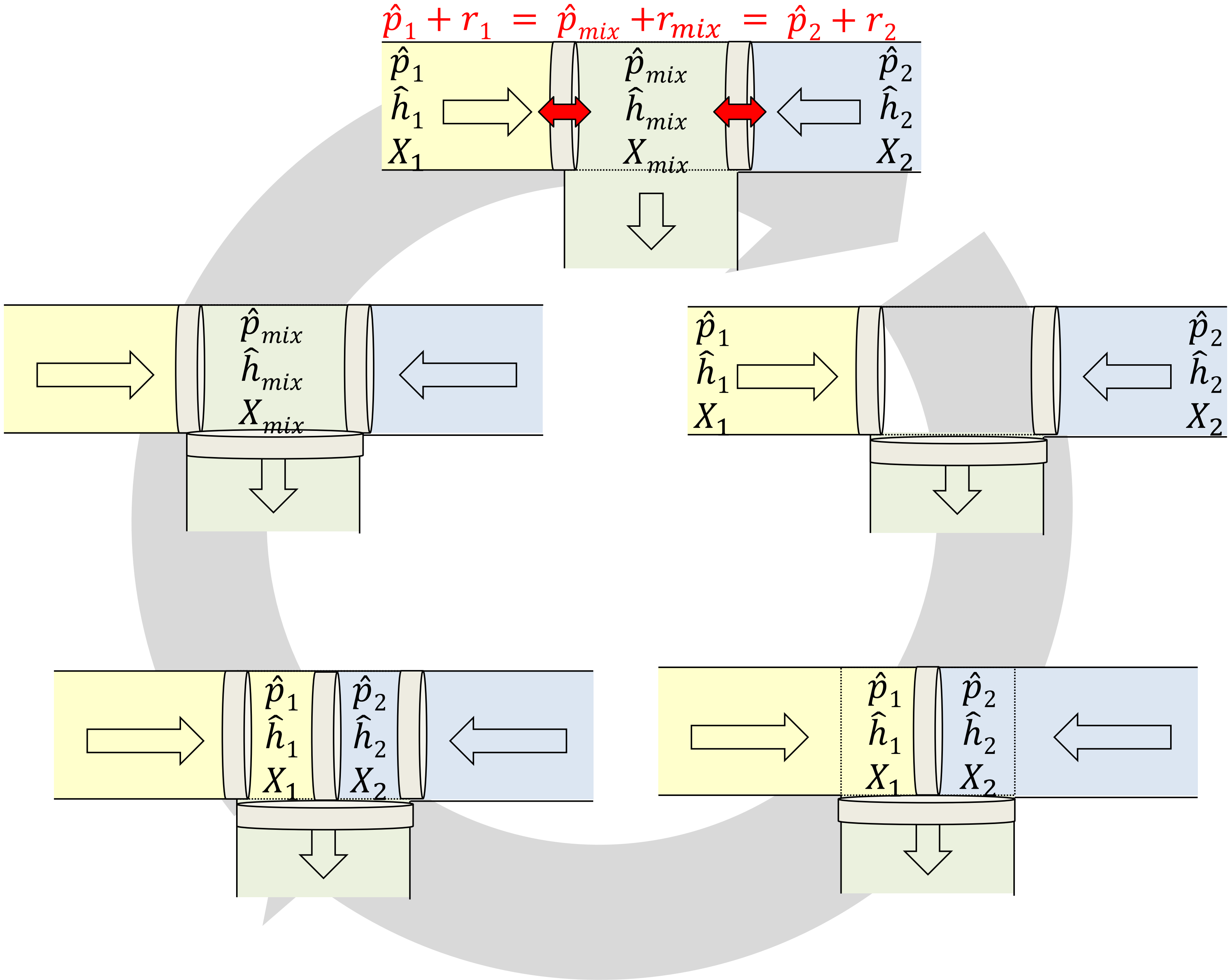


Structure of the resulting equation system



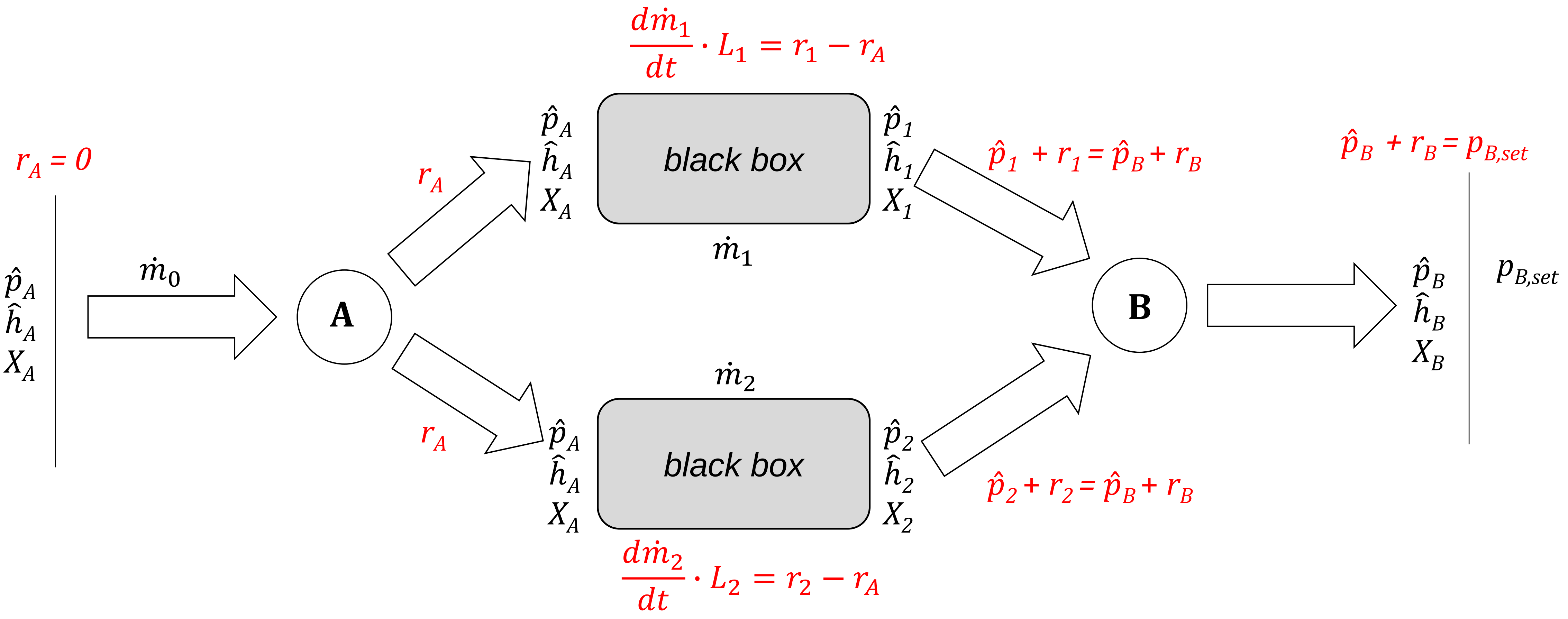
The black variables can be computed downstream

Mixing $\hat{p}$



$\hat{p}_1 + r_1 = \hat{p}_{mix} + r_{mix} = \hat{p}_2 + r_2$
$\hat{p}_1$ $\hat{h}_1$ X_1
$\hat{p}_{mix}$ $\hat{h}_{mix}$ X_{mix}
$\hat{p}_2$ $\hat{h}_2$ X_2

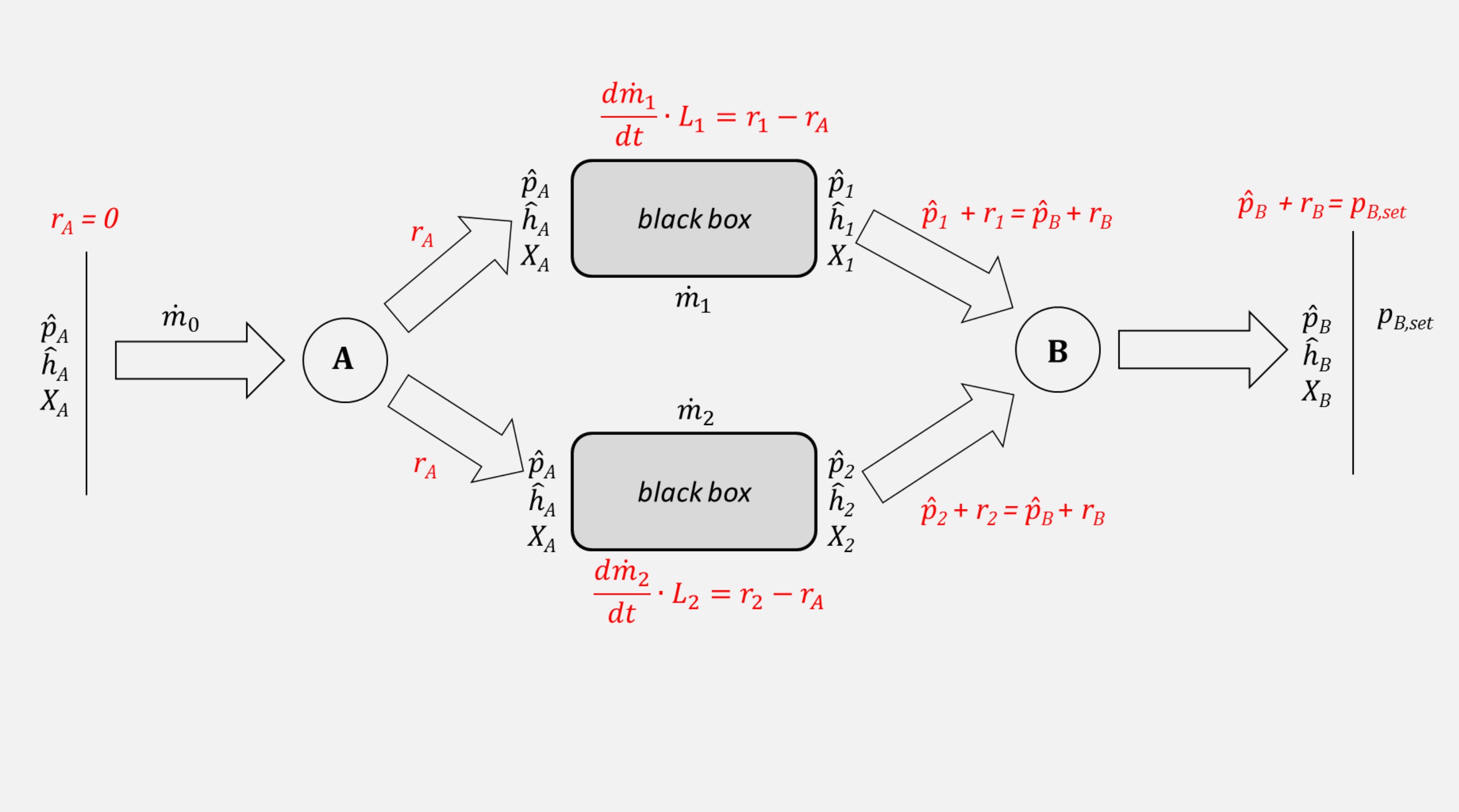
Structure of the resulting equation system



The black variables can be computed downstream

What about the red variables? How to determine: $d\dot{m}_1/dt,\ \ r_1\ d\dot{m}_2/dt, r_2$

Structure of the resulting equation system



We can extract:

- $\frac{d\dot{m}_1}{dt} \cdot L_1 = r_1 - r_A = r_1$
- $\frac{d\dot{m}_2}{dt} \cdot L_2 = r_2 - r_A = r_2$

At junction B:

- $\hat{p}_1 + r_1 = \hat{p}_2 + r_2$

Differentiation splitter A:

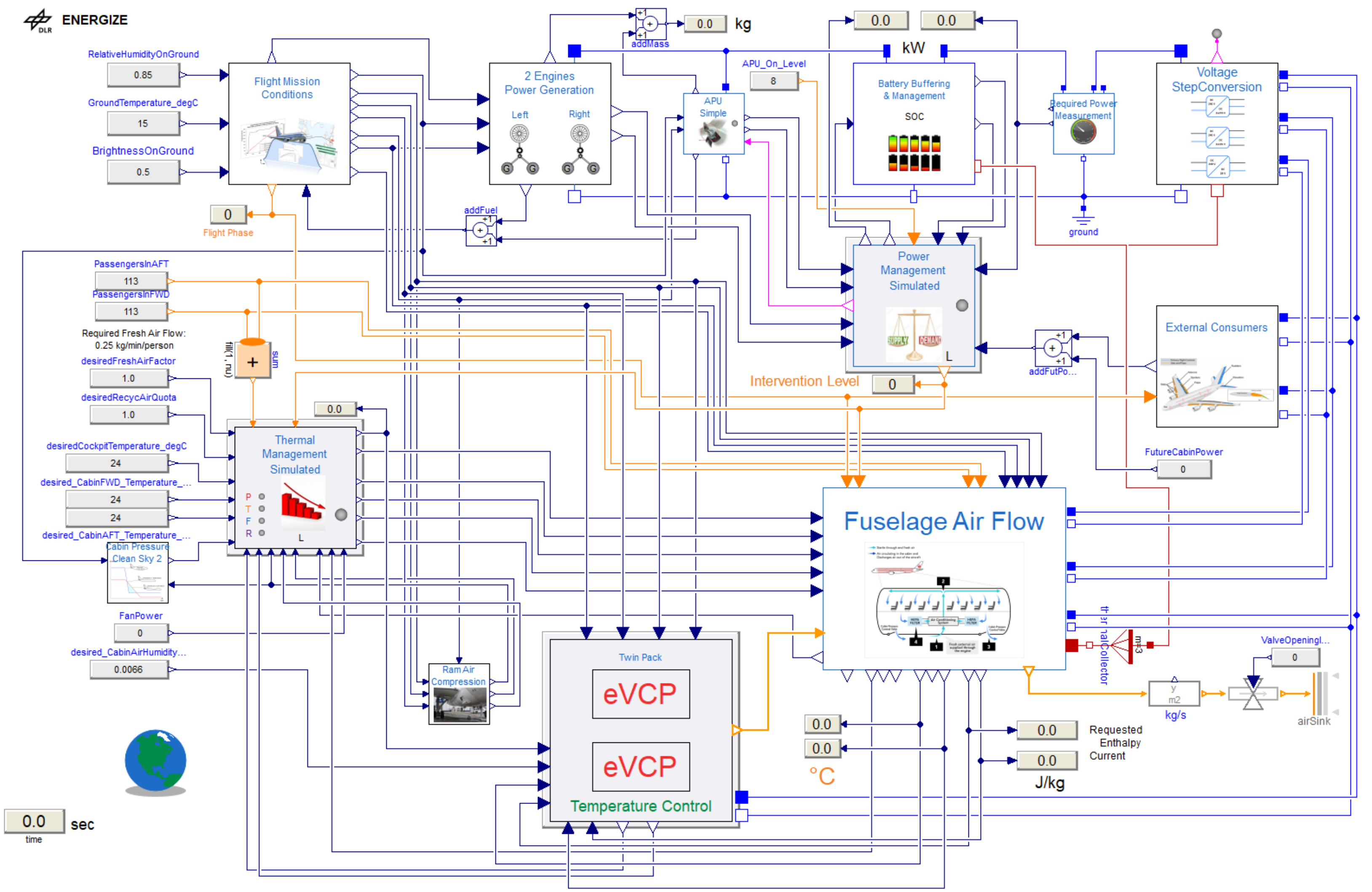
- $d\dot{m}_1/dt = -d\dot{m}_2/dt$

from $\dot{m}_1 + \dot{m}_2 = \dot{m}_0$, $\dot{m}_0$ is given

$$\begin{bmatrix} -1 & & L_1 & \\ & -1 & & L_2 \\ 1 & -1 & & \\ & & 1 & -1 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ d\dot{m}_1/dt \\ d\dot{m}_2/dt \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \hat{p}_2 - \hat{p}_1 \\ 0 \end{bmatrix}$$

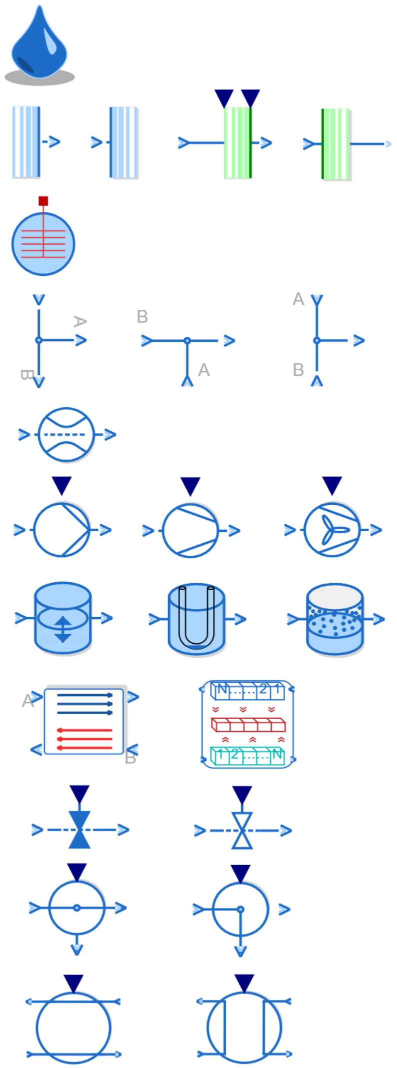
- Linear Equation System
- Matrix with constant elements
- Can be inverted upfront
- Dymola does this automatically

Diese Methode skaliert



- The ENERGIZE Model represents a More-Electric Aircraft with approx. 220 passengers
- Combined electrical and thermal load management.
- Simulation of complete flight missions under different environmental conditions
- > 18,000 Equations
- > 300 States

Open Source Library: DLR ThermoFluid Stream



Commons Model

Source, Sink,
dynamic pressure boundaries

Conduction Element

Splitter / Junctions

Flow Resistance

Pump, Compressor, Fan

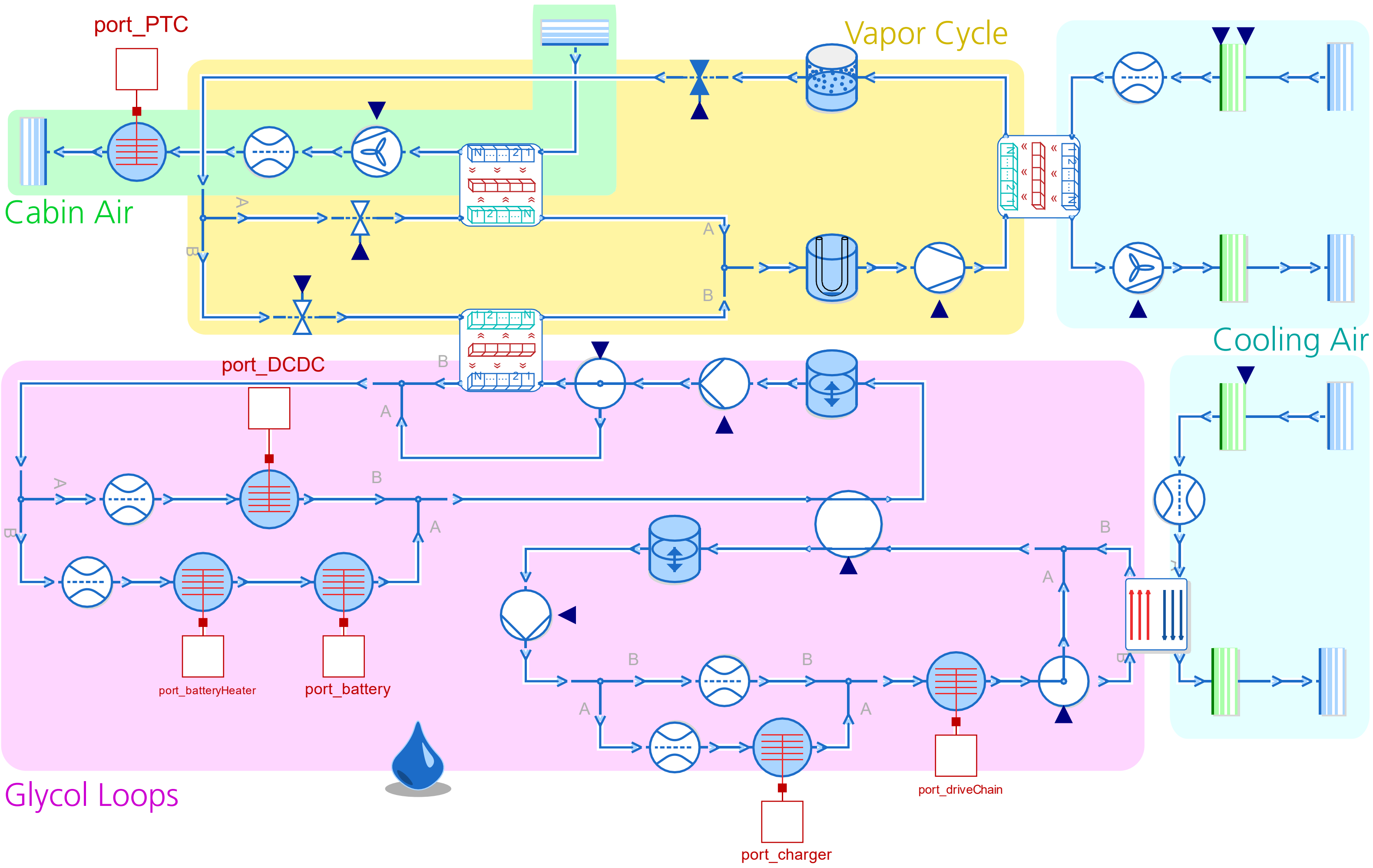
Reservoir,
Accumulator, Receiver

E-NTU Heat Exchanger,
Discretzied Heat Exchanger

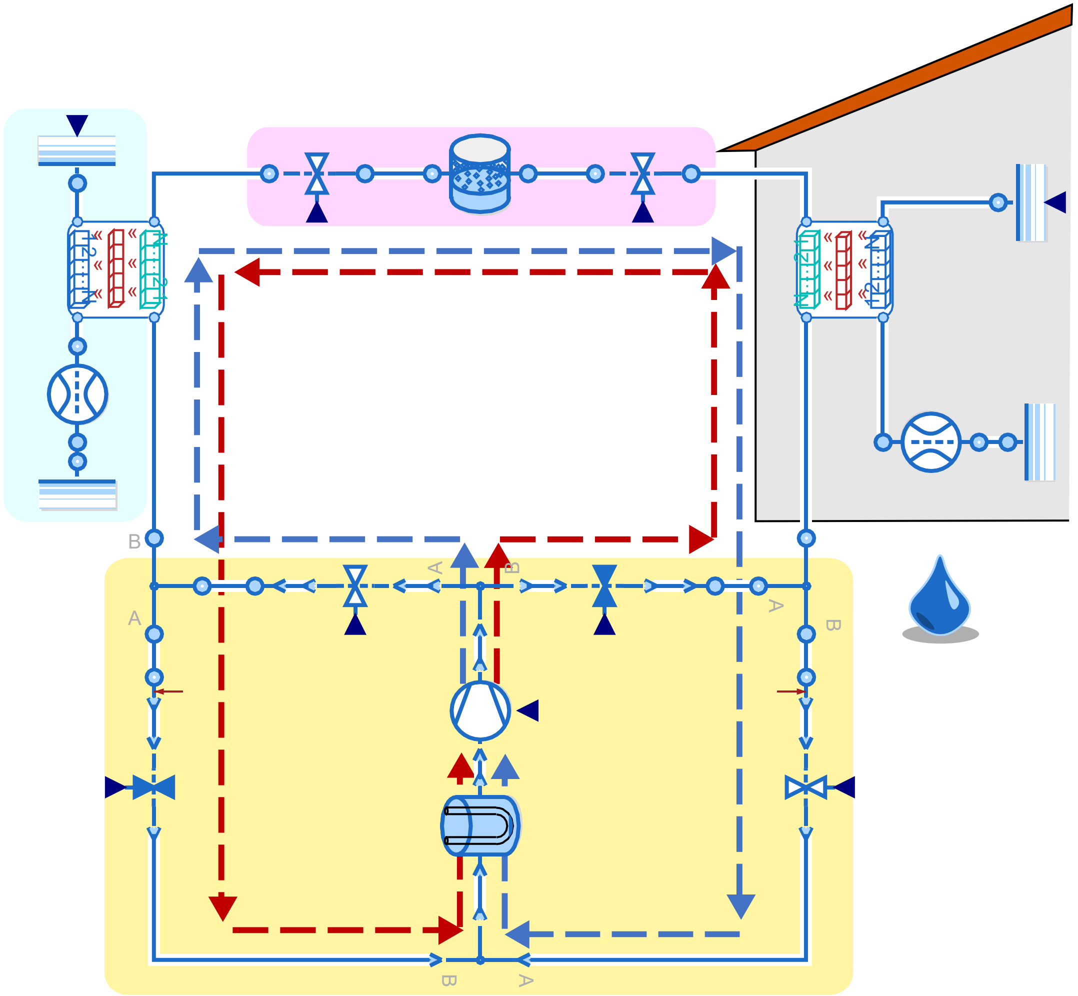
Pressure Control Valve,
Linear valve

Switching Valve

Loop Splitter Valve

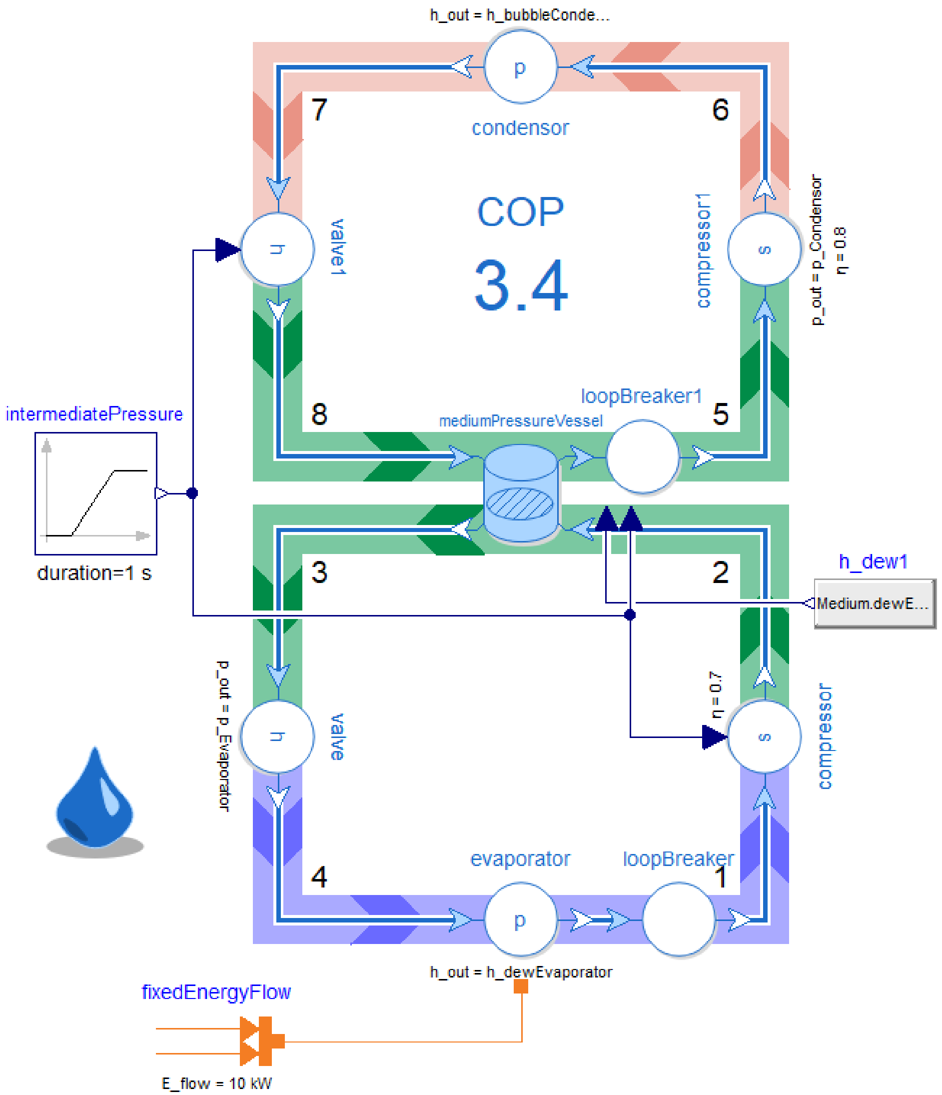


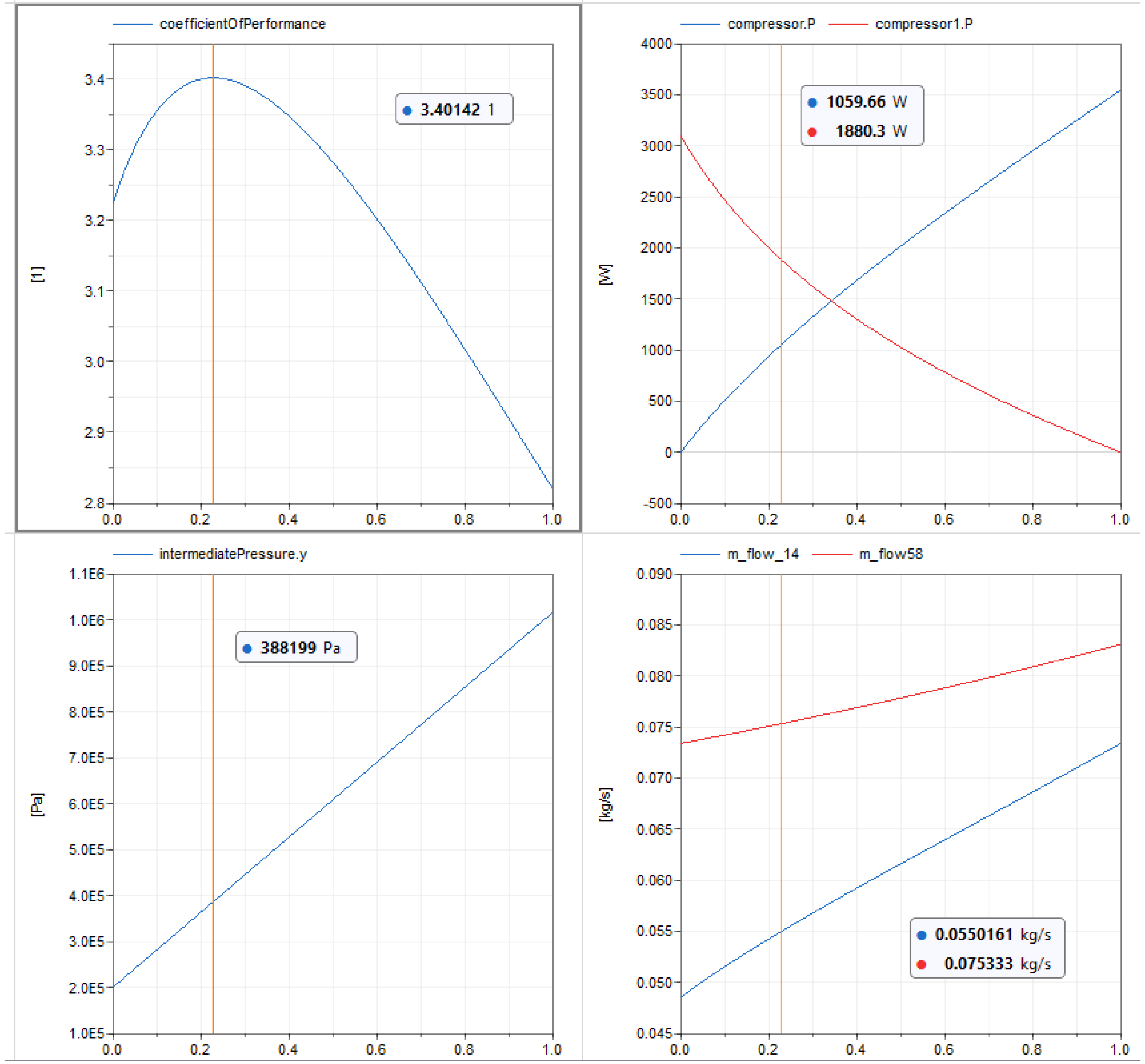
Example of Reversible Heat Pump



- Support for bi-directional components
- Two-Phase Heat Exchanger Models
- Interface for various Media Models (esp TIL Media)
- New Pump Models
- New Models for Pipes based on Idel‘chik

New Development for Ideal Processes





Where to get the library?

Find it on GitHub:
https://github.com/DLR-SR/ThermofluidStream

Feel free to contribute!!

Release	Dymola Version 2021 (64-bit) 2020-04-17	OpenModelica * OpenModelica v1.19.0-dev-38-ge9f86ba1ce (64-bit) OMSimulator v2.1.1.post80-g1bf17f4-mingw	Modelon Impact **
v0.1-beta	✓ (runs and passes regression test)	✗	✗
v0.2-beta	✓ (runs and passes regression test)	(✓) Fully compiles, mostly runs and passes regression test	✓ (runs)
Latest release	✓ (runs and passes regression test)	✓ Fully compiles, mostly runs and passes regression test	✓ (runs)

* Problem with correct setting of assertion level (should be fixed in future release of OpenModelica)

** Not available for testing to us

Comparison to existing Methods

Robustness

Finite Volume Approach:
- no implicit non-linearities
- many states
- stiff
- high frequencies

ODE

DAE

DLR ThermoFluid Stream:
- No implicit non-linearities
- few states
- Stiffness can be manipulated
- Frequency can be manipulated

AE

Algebraic Stream Approach:
- Complex non-linearities in implicit form
- No states or very few
- Not stiff

Performance

LINEAR IMPLICIT EQUILIBRIUM DYNAMICS

Repetition: From Necessary to Sufficient

- Our current standard interfaces.
  They are what is ***necessary*** for object-oriented modeling.

Domain	Translational Mechanics	Rotational Mechanics	Hydraulics	Electrics	Thermal	…
Potential	r	φ	P	V	T	
Flow	f	τ	$\dot{Q}$	i	Q	

- We can find extended interfaces that offer a ***sufficient*** form.
  (Unfortunately hardly anyone is looking for these forms)

Domain	Translational Mechanics	Rotational Mechanics	Thermo Fluids	Electrics	?	…
Potential	v_{kin}	ω_{kin}	r	?	…	
Flow	f	τ	$\dot{m}$	?	…	
Signal	r	φ	$\boldsymbol{\Theta}$	?	…	

Definition of a Linear Equilibrium Dynamics System

- The way of modeling that we derived leads to a special class of DAE systems: Linear Implicit Equilibrium Dynamics.

$$\mathbf{0} = \boldsymbol{F}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{w}, t)$$

$$\begin{bmatrix} \dot{\mathbf{x}}_E \\ \mathbf{w}_E \end{bmatrix} = \mathbf{g}(\mathbf{x}_E, \mathbf{x}_I, t)$$

$$\mathbf{A}(\mathbf{x}_E, \mathbf{x}_I, \mathbf{w}_E) \begin{bmatrix} \dot{\mathbf{x}}_I \\ \mathbf{w}_I \end{bmatrix} = \mathbf{f}(\mathbf{x}_E, \mathbf{x}_I, \mathbf{w}_E, t)$$

The DAE is linear in the state derivatives $\dot{\mathbf{x}}$ and the algebraic variables $\mathbf{w}$

- What looks like a very restrictive class of models is actually much more powerful than expected.

Further Developments

Robust Modeling Libraries for Entry-Level Use

- ThermoFluid Streams
- Mechanics 1D, 2D, 3D
- Electrics
- Controlled Power Flow for Design

New Compilation Mechanism for Modelica

- Better use of Modelica in Teaching
- Avoid Flattening
- Simulation of Large Systems
- New options for NVIDIA, ARM, WebAssembly based on LLVM