Simulating a Pneumatics Network using the DLR ThermoFluidStream Library

ASIM STS, GMMS & EDU Workshop 2025
DLR Oberpfaffenhofen, 11. 4. 2025

Peter Junglas, Raphael Gebhart

Die unternehmende Hochschule für die Region

Content

Introduction

Pneumatic system:

- turbulent flow of a compressible medium
- large pipe network
- highly non-linear components (eg actuators, valves)

Modeling approaches for components:

- PDE-based (CFD)
- 1d/2d coarse grained finite volume (special Modelica libraries)
- 0d = simple ODE or algebraic equation (standard Modelica library MFL)

Modeling of large networks:

- using MFL is cumbersome (Drente, Junglas 2015)
  - large system of nonlinear equations
  - initialisation problem is still unsolved (here)
  - tee branches particularly difficult
  - models only run after drastic simplifications
- DLR ThermoFluidStream Library TFS (Zimmer et al. 2022)
  - generally assumes fixed flow direction
  - adds inertial pressure of fluid → mass flows become state variables
  - clever approximation scheme decouples components
  - leads to linear or small non-linear equations → initialisation often works
  - suitable approach for pneumatic pipe networks?

Using the DLR ThermoFluidStream Library

Basic ideas of the TFS library:

- incorporate pressure difference due to fluid inertia

$$\Delta r = -L\frac{d\dot{m}}{dt}$$

  - often neglected in quasi-static processes (MFL)
  - inertance L independent of thermodynamical state
  - in TFS generally defined as small global constant
- steady mass flow pressure $\hat{p}$ given by

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

  - approximate its change in a component

$$\Delta\hat{p} = f(p, \dot{m}) \approx f(\hat{p}, \dot{m})$$

  - leads to decoupling of components
  - makes initialization problem feasible
- connectors

r	r
m_flow (flow)	m_flow (flow)
state (input)	state (output)

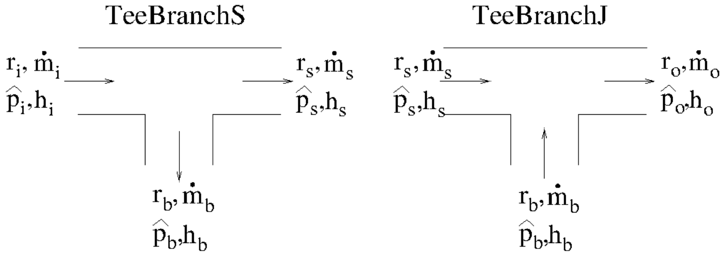
Specialized library PneuBibTFS:

- mainly wrappers around TFS components
  - define medium as SimpleAir
  - reduce number of parameters
- examples
  - Pipe, Bend
  - Tank (with inflow and outflow)
  - PressureSource, PressureSink
- MassFlowSource, MassFlowSink
  - mass flow is state variable in TFS!
  - implemented with control valve using PT1 dynamic
  - simpler with linear valve
  - actuator = linear MassFlowSink
- tee branches need special care

Modeling Tee Branches (1)

Tee branches:

- split or join mass flows



  - simplification: 90° angle and identical cross sections A
- complex behaviour
  - changed total cross sections → dynamic pressure changes
  - internal friction → pressure drops
- nonlinear coupling across the complete model
  - MFL: drastic simplifications necessary (substitutional pipe length)
  - TFS: no nonlinear coupling of components (mass flow is state variable!)

Equations of basic splitter TeeBranchS:

- simplifications
  - constant temperature
  - constant density, coming from input state
  - in applications usually ok
- loss functions ζ_{cs}, ζ_{cb}
  - describe friction and part of dynamical pressure
  - here: simple fit polynomials
- basic equations

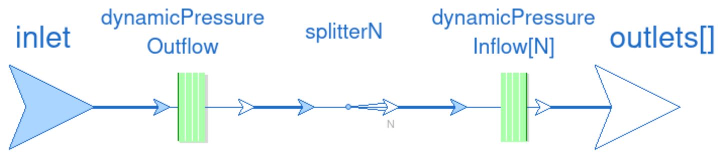
$$
\begin{aligned}
0 &= \dot m_i + \dot m_s + \dot m_b \\
\rho &= \rho(\hat p_i, h_i) \\
\Delta p_s &= -\frac{1}{2\rho A^2}\zeta_{cs}\left(\frac{\dot m_b}{\dot m_i}\right)\dot m_i^2 \\
\Delta p_b &= -\frac{1}{2\rho A^2}\zeta_{cb}\left(\frac{\dot m_b}{\dot m_i}\right)\dot m_i^2 \\
\Delta p_{dyn,s} &= \frac{1}{2\rho A^2}(\dot m_i^2 - \dot m_s^2) \\
\Delta p_{dyn,b} &= \frac{1}{2\rho A^2}(\dot m_i^2 - \dot m_b^2) \\
\hat p_s &= \hat p_i + \Delta p_{dyn,s} + \Delta p_s \\
\hat p_b &= \hat p_i + \Delta p_{dyn,b} + \Delta p_b \\
h_s &= h_i \\
h_b &= h_i
\end{aligned}
$$

- in TFS additional equations for r variable (Zimmer 2020)

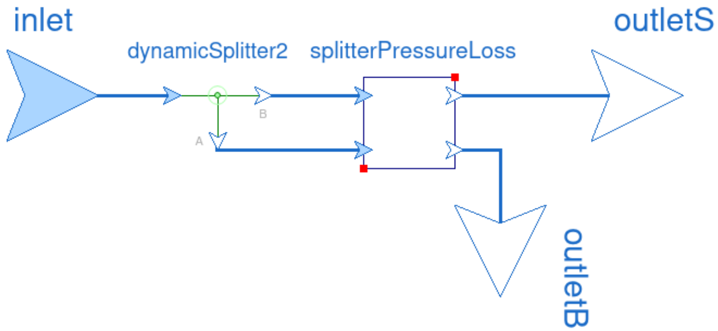
Modeling Tee Branches (2)

Variant TeeBranchS1:

- uses DynamicSplitter from TFS



  - computes dynamic pressure differences using cross section areas
- adds SplitterPressureLoss component for ζ functions



- differences to TeeBranchS
  - DynamicSplitter is adiabatic, not isothermal
  - SplitterPressureLoss uses density after DynamicSplitter, not at Inlet
  - 147 equations instead of 28

Variant TeeBranchS2:

- basically like TeeBranchS1
- DynamicSplitter uses densities at input and output streams
- adds two nonlinear equations inside the component

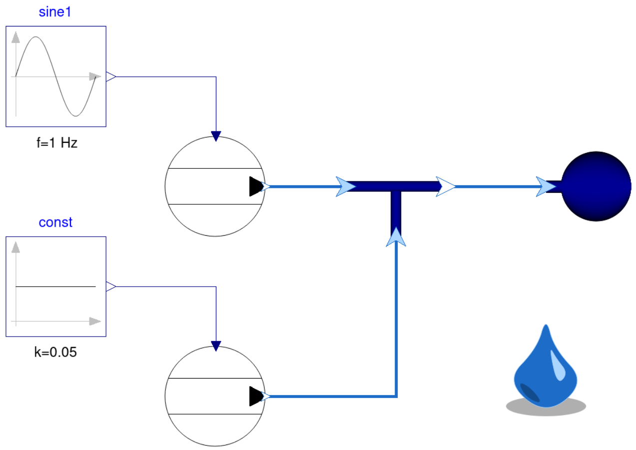
Joining components:

- TeeBranchJ, TeeBranchJ1 and TeeBranchJ2, similar to splitters
- different handling of r variables

Testing Tee Branches (1)

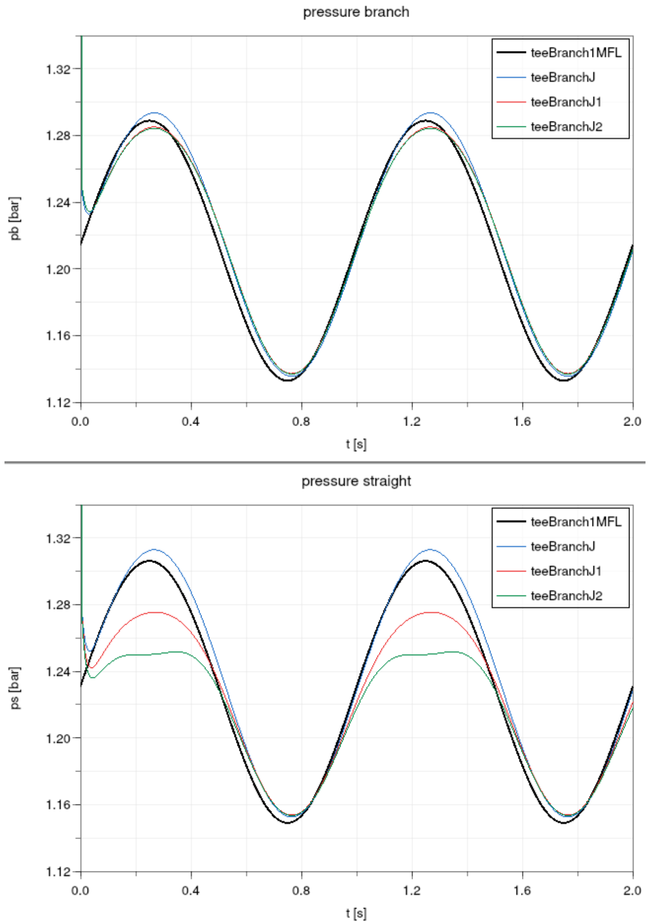
Basic test models:

- join case, mass flows given at inlets, pressure at outlet



- results for basic components in MFL and TFS almost identical
- deviations in straight branch for density aware components

- results for MFL and several TFS components



Testing Tee Branches (2)

Stability of models:

- possible behaviour
  - runs with standard initial conditions
  - runs with special initial conditions
  - doesn't run
- tests with different boundary conditions
  - pressure at inflow, mass flow at outflow
  - mass flow at inflow, pressure at outflow
  - pressure at inflow and outflow
- results
  - MFL model works always
  - all TFS models work for simple cases (two mass flows given)
  - basic TFS component works in most non-simple cases
  - other TFS components work never in non-simple-cases
- problem
  - initialization works always
  - pressure soon rises exponentially
  - differential equations are highly unstable!
  - thorough mathematical investigation needed

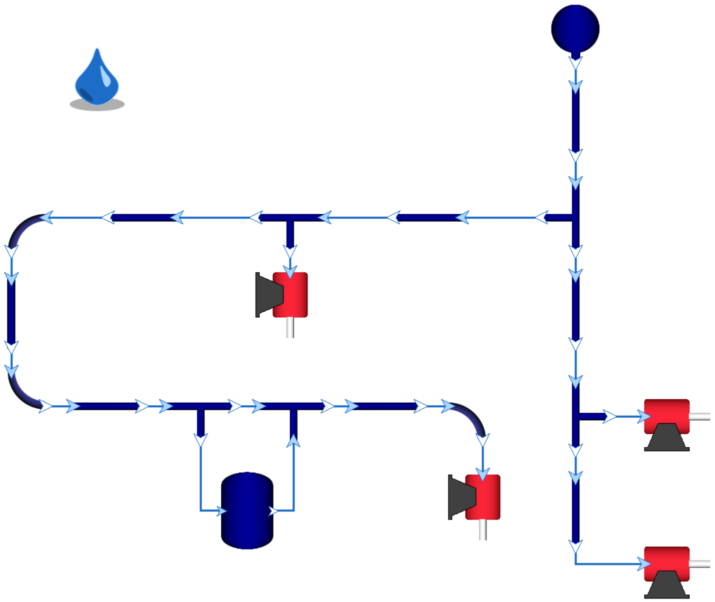
Adding a pipe at straight inflow or outflow:

- simple cases
  - all TFS work (as before)
  - some MFL models don't work!
- non-simple cases
  - non-conclusive
  - some MFL models get unstable (don't run)
  - some TFS become stable
- preliminary conclusion
  - adding pipes destabilizes MFL (larger nonlinear system)
  - adding pipes stabilizes TFS

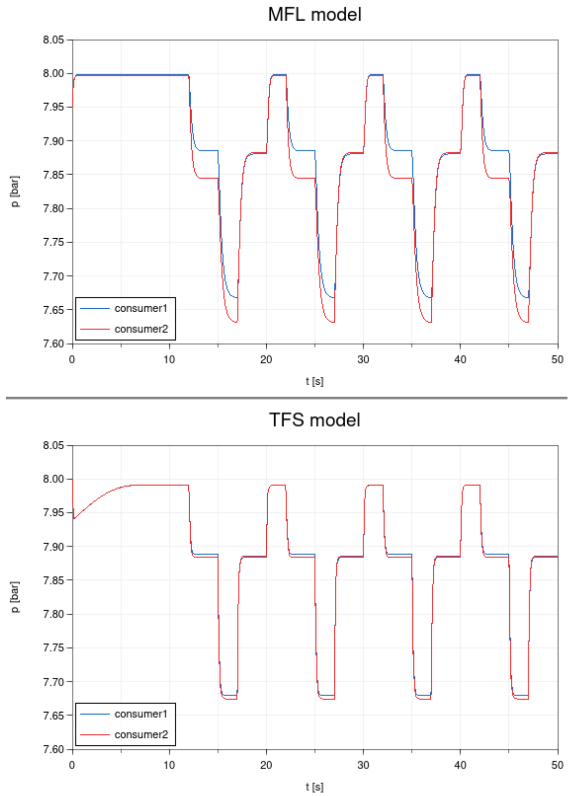
Modeling Pneumatic Networks (1)

Example model:

- network 1



- results



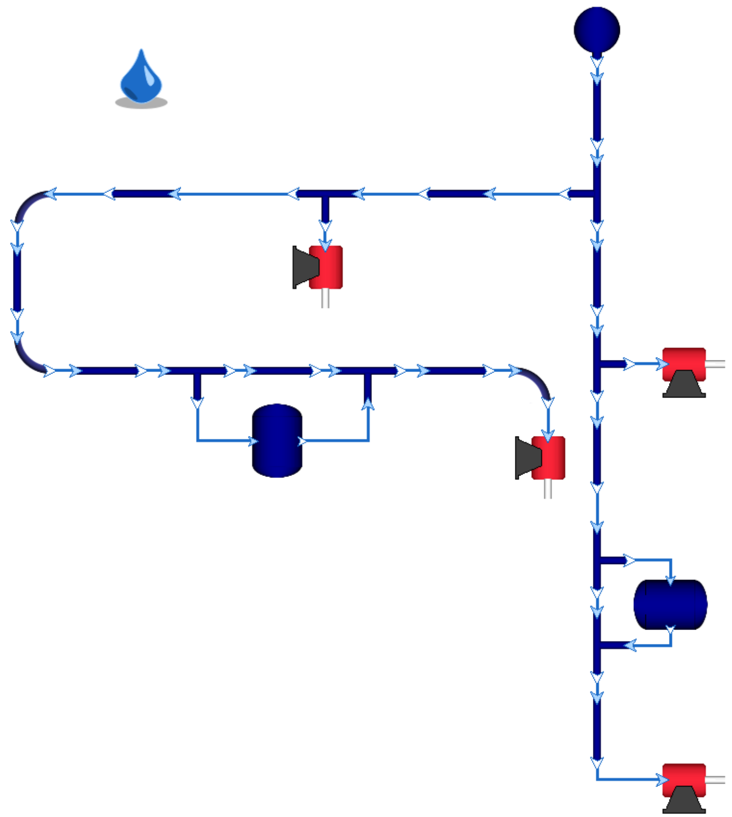
- difference to MFL version
  - tank has dedicated inflow and outflow
  - connected via a loop with splitter and joiner
- stability
  - MFL: only runs with simplistic tee branch models
  - TFS: all three versions run with standard initial conditions
  - → preliminary conclusion from tests verified

  - differences between tee branch versions below plot accuracy
  - negligible temperature and density changes

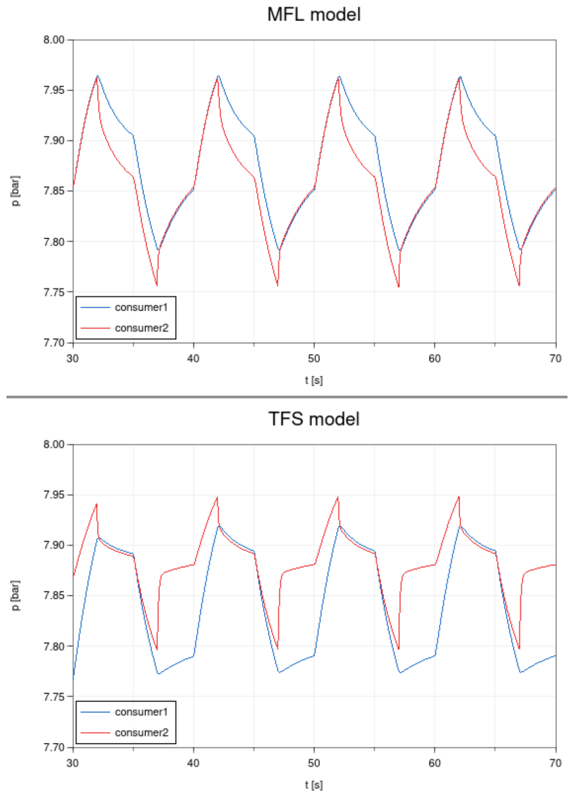
Modeling Pneumatic Networks (2)

Extended example model:

- adds auxiliary tank betwen two consumers



- results



- stability
  - base model stops at 80 s (consumer at tank 1 is switched on)
  - simple remedy: add small pipe in tank loop

- significant deviations due to drastic simplifications in MFL version
- smaller pressure variation at consumer2

Modeling Pneumatic Networks (3)

Real-world model:

- medium-sized model from industrial partner
  - implemented in PneuBib + MFL
  - 60 components (3 pumps, 1 tank, 12 consumers, 17 simple tee branches)
  - ≈ 4500 equations
- porting to PneuBibTFS
  - specify flow directions
  - include tank via a small loop
  - initial pressure of tank = pump pressure (all equal)
  - ≈ 1750 equations
- simulation stops after 1 s
  - several flows have wrong direction
  - due to identical pump pressures
- remedy
  - increase pressure of one pump marginally
  - → all flow directions fixed
  - → model runs, reproduces results of MFL version qualitatively

Conclusions

TFS-based pneumatics library:

- usually works directly
  - in case of instability add auxiliary pipes
- more accurate results
  - dynamic pressure changes missing in crude MFL version
- more detailed tee branch components
  - change results only marginally
  - generally reduce stability

Using OpenModelica:

- introduced additional problems with MFL version
- with PneuBibTFS no differences to Dymola
  - enhanced OpenModelica simulator
  - TFS models much simpler for simulator

Enhancement of tank model:

- more realistic: use one tank port in both flow directions
- possible due to TFS enhancement for bidirectional flows
- open questions
  - significantly better results?
  - stability?

Problem of stability:

- cause
  - MFL: nonlinear equations
  - TFS: unstable differential equations
- probably related
- open questions
  - mathematical analysis of instability
  - mechanism of stabilization through pipes

Initialization problem:

- in MFL still open (Drente, Junglas 2015)
- in TFS solved (at least for pneumatics networks)