Simulating a simple pneumatics network using the Modelica Fluid library

Patrick Drente[1], Peter Junglas[2]
[1]Waskönig+Walter Kabel-Werk GmbH u. Co. KG
[2]PHWT Vechta/Diepholz/Oldenburg
peter@peter-junglas.de

The Modelica Fluid library provides a framework and a large number of components for thermo-fluid applications. This should simplify the task of modeling a large pneumatics system considerably. Nevertheless a lot of problems remain. Some of them are due to the lack of important components that are difficult to model, others are related to deficiencies of the used modeling software. But the really hard problems come from conceptual difficulties that are well known to experts and plague the practical modeler. The lessons learned from building and using a basic pneumatics library show what is feasible right now - and where future work is needed to make modeling of fluid systems easier.

1 Introduction

Pneumatic systems are used in industrial applications to distribute power in factory buildings. Such installations can be huge, consisting of large distribution networks, several compressors and many different kinds of consumers. Planning a new network or optimising an existing one involves a large number of parameters and possible configurations. A particular problem is the unknown timing behaviour of the consumers, which is often highly irregular. This obviously calls for an appropriate simulation tool.

The simulation of fluid systems with compressible media is a difficult task. A large step forward has been the introduction of the Modelica Fluid library [1] (in the following often abbreviated as MFL). It incorporates the fundamental behaviour of one-dimensional thermo-fluid systems and contains basic components for vessels, pumps, valves, pipes and other network elements. It uses the Modelica Media library [2], which allows to choose from a large list of predefined fluid media, compressible as well as incompressible.

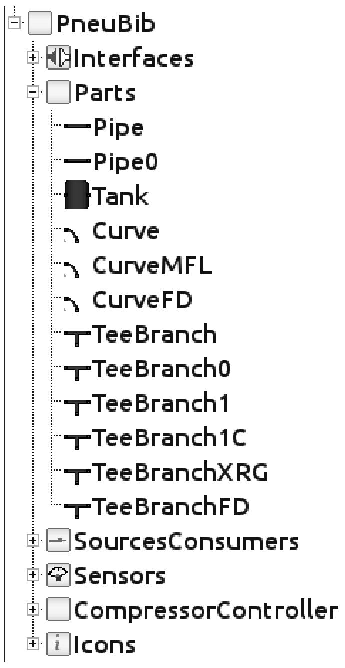
Under these conditions Waskönig+Walter decided to start a simulation study in cooperation with the PHWT to optimise its existing pneumatics network. The simulation should help to find the reasons of bottlenecks and to evaluate the effects of simple actions beforehand. The tool chosen was OpenModelica, which is open source and has a good support of the MFL. From the academic point of view two questions were of special interest: How easy is it to use the MFL for development of own libraries? What is the current status of OpenModelica concerning MFL?

In a first step a simple library has been constructed that contains all necessary components. It does not compete with any of the commercial pneumatics libraries available, but concentrates on the basic network. Notably missing are models for valves and actuators. A special feature is a focus on the tee branch, which is a common device in pneumatics networks and is astonishingly hard to model with reasonable accuracy.

The choice to work with OpenModelica had serious consequences. To make clear which of the difficulties are intrinsic to the modeled system and which are due to deficiencies of the software, the construction of the library and the models will be described in the following using Dymola, which is arguably the best platform for MFL applications at present. The particular problems coming from using OpenModelica are subject of a later section.

2 The PneuBib library

All components that are needed in the following to model simple pneumatics networks are combined in the PneuBib library. Two basic assumptions are made throughout: The temperature is constant and given by the ambient temperature, and the medium used is SimpleAir from the Modelica.Media library.



PneuBib
- Interfaces
- Parts
  - Pipe
  - Pipe0
  - Tank
  - Curve
  - CurveMFL
  - CurveFD
  - TeeBranch
  - TeeBranch0
  - TeeBranch1
  - TeeBranch1C
  - TeeBranchXRG
  - TeeBranchFD
- SourcesConsumers
- Sensors
- CompressorController
- Icons

Figure 1: PneuBib library

The model names are in German, but will be translated here for better readability. The library consists of the following packages (fig. 1):

- `Interfaces` and `Icons` provide the common infrastructure of ports, base classes and functions.
- `Parts` contains the main components for the network and will be discussed in more detail below.
- `SourcesConsumers` includes the usual source blocks to define pressure, mass or volume flow as well as a generic consumer block. This is simply a wrapper around the linear valve component from the MFL connected to the ambient pressure. For convenience a special consumer is added that opens periodically with additional parameters for number of periods and start time.
- `Sensors` provide access to the values of pressure, mass or volume flow in a network model.
- `CompressorController` contains components for the modelling of the controlled compressors that are used at Waskönig+Walter.

The `Parts` sublibrary contains a few components that are wrappers around corresponding MFL blocks: The `Tank` is an isothermal version of the ClosedVolume, the `Pipe` consists mainly of a DynamicPipe with two nodes. The `CurveMFL` uses the CurvedBend from the MFL Fittings package. Unfortunately it doesn't work in OpenModelica, therefore a simpler version `Curve` has been added that relies only on the MFL function dp_curvedOverall_DP to compute the pressure loss.

Finally `Parts` provides several versions of a tee branch. They are the most complex part of PneuBib and will be explained in the following section.

3 Modeling a tee branch

The seemingly simple tee branch is difficult to model due to its several operational modes corresponding to the directions of the flows at its three ports: It can be used for splitting the main flow using the side branch either as one of the outgoing directions or for the incoming flow. One can join two flows with the combined outgoing flow either going straight or through the side branch. And for a compressible medium one could even use all three connections in the same direction, either outgoing or incoming.

The flow situations in all these cases are completely different – and always very complicated. Fortunately we are not interested in the exact flow but only in the overall pressure drops. Of course these depend on many details like the exact geometry of the pipes or the roughness of the inner pipe surfaces. But for our purpose of designing or analysing a pipe network, a simple approximation is often good enough. For this reason the PneuBib library contains several tee branch models with different levels of complexity and accuracy.

The MFL contains two models named `TeeJunctionIdeal` and `TeeJunctionVolume`, but they are not useful here, since they only describe the mixing properties, not the pressure loss in the junction.

Fortunately this gap has been filled by the free package FluidDissipation [3], which contains a lot of functions for heat transfer and pressure loss in many important cases, among them the tee branch. For the computation of pressure losses the library relies heavily on [4], a large compilation of knowledge in form of formulae, tables and graphs. To implement these into numerically stable models they had to be complemented by a whole bunch of interpolation and regularisation formulae and combined with sophisticated schemes to discriminate between the different modes of the tee junction.

In addition to the necessary functions the FluidDissipation library contains a ready-to-use example component of a tee branch. This has been incorporated into the PneuBib as `TeeBranchFD`, which adds only a wrapper to fix many of the parameters that are not used in the pipeline context. The `TeeBranchXRG` component is a simplified version that uses the pressure loss functions directly.

A simpler version is `TeeBranch1`, which is reduced in three ways: First it implements only the two modes that are used in the following, namely splitting and joining along the straight direction. Second, it neglects all density changes and uses only the density at the straight input for all computations. And finally it computes the pressure drops by using only simple interpolation polynomials for the pressure drop coefficients ζ_i as functions of the volume flow ratio $Q_{\text{branch}}/Q_{\text{combined}}$. The task of finding an appropriate polynomial is not easy – not because it is hard to find one, but because there are many published versions. Fig. 2 gives an impression of the large variations in published values. The relations that are implemented in `TeeBranch1` are based on data given in [5] for the split case and on the formulae in [6] for the join case. Fortunately the general conclusions for the models of interest here do not depend significantly on the details of the chosen curve.

The largest impact of the simplifications has the assumption of constant density. Therefore PneuBib contains the variant model `TeeBranch1C` that uses the appropriate variable densities for computing energy balances. Only for the computation of the ζ values it sticks to the mentioned polynomials – mainly because

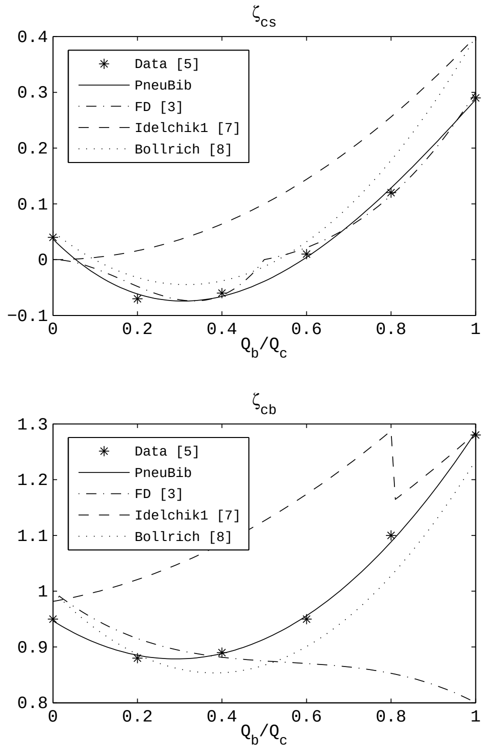


Figure 2: Pressure drop coefficients for the splitting tee branch

better results for compressible media are not available.

When dimensioning pneumatic networks in practice one often uses a very basic approach to include the pressure losses of tee branches: At each outgoing junction one adds a “virtual” substitutional pipe that reproduces the pressure loss of the tee branch [9]. By choosing an appropriate pipe length according to the dimensions of the junction one can get a rough approximation of the pressure losses. Values for the substitutional length can be found from vendors of pneumatic equipment, e. g. at [10]. The basic component `TeeBranch` implements this idea.

4 Testing the tee branch components

Several tests have been performed to check the basic functionality and to compare the different models of the tee branch. Fig. 3 presents one of the test models.

It is used to measure the pressure drops in the joining case, where a given time varying mass flow is inserted at the left port and a constant mass flow at the side port. The outgoing port on the right is connected to a fixed pressure.

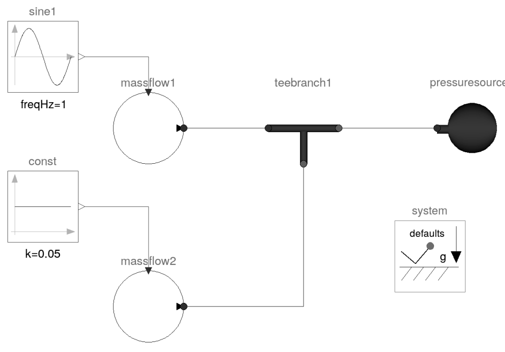


Figure 3: Model for testing a tee branch component

The resulting pressure drops from the straight and the side connection to the output are compared in fig. 4 for the three models `TeeBranchFD`, `TeeBranch1` and `TeeBranch`. The curves show a good qualitative agreement, their differences are to be expected considering the variability of the different underlying data.

One important thing that has been learned here is how to choose the parameters for the basic `TeeBranch`: The values given for the substitutional length vary by more then 25 % (e.g. between [9] and [10]), the roughness k of the substitutional pipe is not given at all. But the test results have shown that the roughness has a considerable influence. Therefore the lengths have been chosen according to [10], which includes values for the straight direction as well, and the value for k has been adapted to reproduce approximately the results of the other components. The resulting value of $k = 0.25$ mm seems reasonable being in the range for used steel pipes that is given in [11].

Similar tests in the split case have an unexpected behaviour: The elaborate models show an increasing pressure in the straight direction, whereas the simple `TeeBranch` gives a pressure drop. A second thought explains this phenomenon: The pressure rise is the dynamical result of the velocity drop due to the splitting of the flow, an effect that is not incorporated in the simple "substitutional pipes" model of `TeeBranch`.

But this doesn't make the simple model useless, there are two different ways how to cope with it: First one

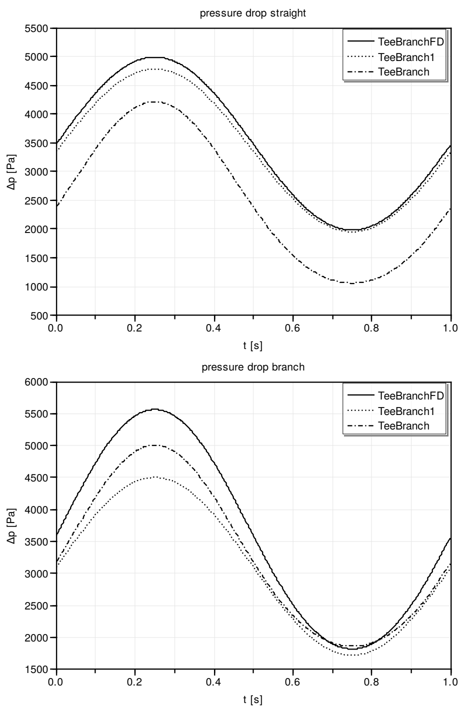


Figure 4: Comparison of the pressure drops in join mode

could just use it, including the pressure drop. After all the substitutional lengths (for the split situation!) are a reasonable rule of thumb coming from practical experience. Most likely it includes dissipative effects that have been neglected here so far. If one wants to reproduce the results of the other tee branch models instead, one can get a pressure rise just by using a negative length of the substituonal pipe. It is rather unexpected that this simple idea works with Modelicas DetailedPipeFlow model, but so it does! In the following the `TeeBranch` will be used with parameter values that roughly reproduce the results of the other models.

To see how the different tee branch models perform as part of a network, in the next test one short straight pipe is inserted between the tee branch and the pressure source. This leads to rapid pressure oscillations with amplitudes of several bar, which are strongly damped and converge to the expected result after a few seconds. Their origin can be easily traced back to the start values: The `System` component defines a default start pressure for all blocks that is set initially to the ambient pressure of about one bar. Changing this to the value given by the pressure source reduces the amplitude of the oscillations to 0.2 bar. Setting the initital mass flow through the pipe to its steady-state value the amplitude goes down to 0.03 bar, which is in the order of the expected pressure losses. To get rid of the remaining oscillations too one had to supply more precise initial pressure values at all three ports, which are generally not known beforehand.

For the final test the position of the pipe has been changed: It is now inserted at the opposite side of the tee branch, directly connected to the mass flow source. In this case the simulation stops after a very short time with an error: The Newton solver is not able to produce reasonable initial values. Trying to fix more initial conditions doesn't help at all, apparently very precise values are needed here for the solver to converge. The problem is the same for all tee branch models in PneuBib, even for the simple `TeeBranch`. Only for the `TeeBranch` with positive substitutional pipe lengths the solver works fine and the results are as expected, but of course with a pressure drop along the tee branch instead of a rise.

A way how to cope with difficult initialisation problems has been proposed in [12]: One substitutes the problematic component with a simpler version that is used only for initialisation. Using a homotopy, i.e. a continuous path from the simple to the complete model, one adapts the initial values gradually, until proper values for the final model are found. This method has been applied successfully in different contexts, especially for thermo-fluid models [13].

According to this idea several simplified versions of the model have been developped, using a heavily reduced tee branch component, a pipe with a linear pressure drop law, a smoothly rising mass flow, a pressure source instead of one flow source or combinations thereof. All of them worked fine without the additional pipe, none of them lead to converging of the Newton solver. The only remaining tee branch model that actually works is the simple `TeeBranch` with positive substitutional pipe lengths.

5 Simulating complete networks

After the basic PneuBib library has been built and tested one can finally turn to the modeling of concrete pneumatics networks. A basic example consisting of several pipes, curves and branches together with a controlled compressor, a few consumers and an auxiliary tank is shown in fig. 5. According to the findings of the last section all branches are simple `TeeBranch` components with positive substitutional lengths given by [10]. If one replaces just one of them with one of the more elaborated models the solver can't find initial values.

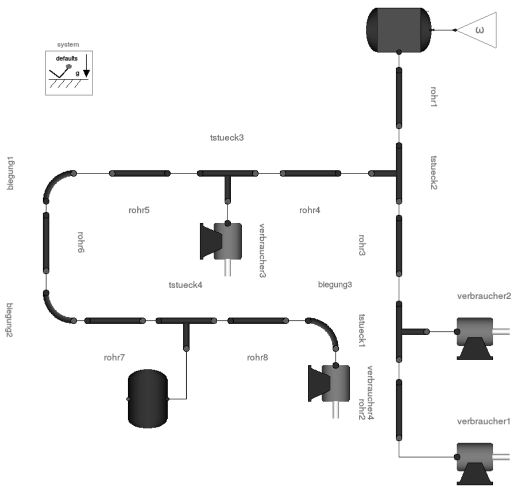


Figure 5: Model of a pneumatics network

Nevertheless such a model can be used to answer some of the questions that appear in real networks, e.g.:

- How large are the pressure drops in several parts of the network and where are the bottlenecks?
- Can the compressor provide the necessary mass flows everywhere, especially in the case of two neighbouring consumers with large demands?
- What size and position should auxiliary tanks have to buffer peak demands?

As a concrete example the situation at the two consumers on the lower right of fig. 5 is considered. Their timing behaviour is shown in the upper graph of fig. 6: Both are used periodically with the same frequency, but different start time, leading to a small overlapping period, where both are working simultaneously. The diagram in the middle of fig. 6 displays the resulting pressures at both positions: When only one is used the pressure drops by about 0.15 bar, and it goes down by almost 0.4 bar, when they are working both. To resolve this problem an auxiliary tank can be installed between the two consumers. The resulting behaviour can be seen in the lower diagram of fig. 6: The total pressure drop is now reduced to almost half of the previous value.

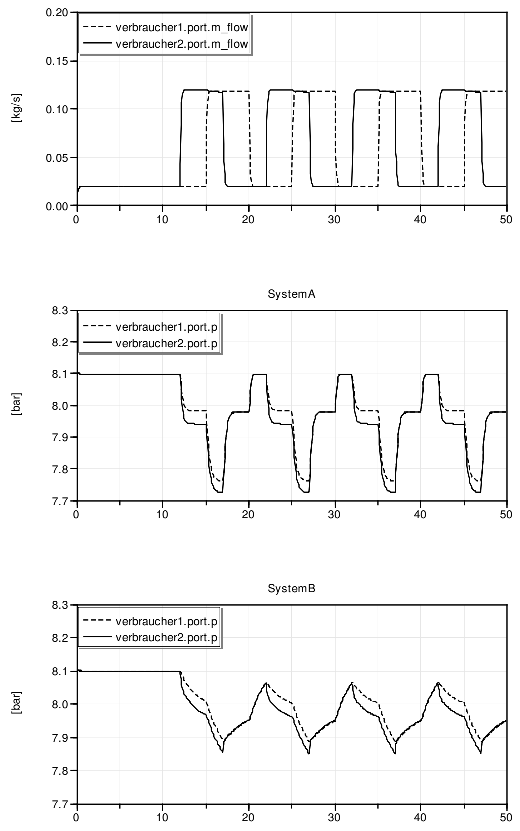


Figure 6: Simulation results of the example network

Large systems with more than 60 components and 5000 equations have been studied in this way. In spite of the severe limitations of the tee branch model their results have been used successfully to improve a real pneumatics network.

6 Working with OpenModelica

As has been mentioned in the introduction all simulations should be performed with the open source program OpenModelica. Though its user interface is not as elaborated as those of commercial programs, its Modelica engine works generally very well in a wide range of applications [14]. Furthermore in the 1.9.1 release, which is the base of this investigation, it provides much better support of the Modelica Fluid library than most commercial programs except for Dymola. For this reason it seemed to be a cheap but promising alternative.

First tests using only MFL components showed promising results, most of the corresponding "wrapped" PneuBib components like `Tank` and `Pipe` worked as well. Only the `Curve` showed a strange behaviour, the simulation lead to undefined values for the mass flow. Using instead the corresponding component `BendFlowModel` from the FluidDissipation library made things worse: Now the simulation lead to an error at initialisation. A working model could be constructed by simplifying the MFL version: Instead of using the function `dp_curvedOverall_MFLOW` to compute the mass flow from the pressure difference and taking into account the different densities and viscosities at both ends, it uses the inverse `dp_curvedOverall_DP` and the density and viscosity only at one end. This worked in OpenModelica and lead to almost identical results in all cases of interest here (as verified later using Dymola).

The real challenge was to find a tee branch model that works in OpenModelica. Tests with the example component that is included in the FluidDissipation library did not succeed, the solver produced an error while flattening the equations. Simplifying this model by using the pressure loss functions directly did not remove the problem. Only after reducing the complexity largely one arrived at a working component that is included in PneuBib as `TeeBranch1` and described above. That the structure of this model is on the brink of OpenModelicas capabilities becomes apparent, when one tries to remove some of its limitations: Using `TeeBranch1C` that includes some effects of the varying density, the simulation stops after a short time and produces very strange results that are due to completely wild initial conditions.

Since due to the initialisation difficulties all the problematic components could not be used in the final models anyhow, the prospects were good that the real pneumatics network could be analysed. Unfortunately in the case of the complete model with more than

100 components and 10000 equations OpenModelica gives up due to sheer size, the Modelica compiler crashes in an early phase. Several simplified versions have been tried with nonconclusive results: Some models with 80 components and 7000 equations run for a short simulation time, before the compiler stops with an error, other much smaller models crash immediately. A reduction of the network to 60 components and 5000 equations has lead to a model that generally runs long enough to produce interesting results even after several structural modifications or parameter changes. It was the basis of the final investigations, which lead to improvements of the real pneumatics system.

7 Conclusions

The modeling and simulation of the “simple” pneumatics network turned out to be much harder than had been expected at the beginning. This is a consequence of several different problems:

- The MFL components are very complex, they contain many parameters and subsystems that are difficult to understand for an unexperienced user. This problem gets much worse if one tries to create similar components from scratch.
- Some important components are missing, especially for the tee branch. The FluidDissipation library is a useful addition here, but it is not easy to use either. Especially the documentation and presentation of ready-to-use components leaves room for improvement.
- The fundamental problem of initialisation seems to be still far from being solved. Maybe the homotopy method is a feasible approach, but at least the authors were not able to find a working solution here.

The decision to use OpenModelica did not help either: Though it generally works fine in simple standard situations, it still has serious problems with models that are very complex or very large. Especially it did not cope well with the FluidDissipation library. In combination with the intrinsic problems listed above this lead to endless debugging sessions where it was almost impossible to isolate the reason of a specific failure.

Nevertheless in the end it all worked out: The industrial partner has now a working tool that helps to optimise his pneumatics installation. It has a sufficient accuracy and is open source. This miracle came about by simplifying the original system dramatically until all difficulties disappeared. Along the way one got rid of the limitations of the tool.

Is the MFL ready for end users? Principally yes – but only, if you are willing to accept simplified solutions and are prepared to work very hard!

References

[1] R. Franke, F. Casella et al. *Standardization of Thermo-Fluid Modeling in Modelica.Fluid.* Proc. 7th Int. Modelica Conference, Como, p. 122-131, 2009.

[2] F. Casella, M. Otter et al. *The Modelica Fluid and Media Library for Modeling of Incompressible and Compressible Thermo-Fluid Pipe Networks.* Proc. 5th Int. Modelica Conference, Vienna, p. 631-640, 2006.

[3] T. Vahlenkamp, S. Wischhusen. *FluidDissipation for Applications – A Library for Modelling of Heat Transfer and Pressure Loss in Energy Systems.* Proc. 7th Int. Modelica Conference, Como, p. 132-141, 2009.

[4] I. E. Idelchik. *Handbook of hydraulic resistance.* Jaico Publishing House, Mumbai, 3rd ed. 2006.

[5] E. R. Rodriguez. *Technische Hydraulik.* Vorlesungsskript, FH Frankfurt. Online: http://slideplayer.org/slide/209801/ (Aufruf 2015-04-15).

[6] D. Aigner. *Gleichung zur Berechnung der hydraulischen Verluste der Rohrvereinigung - kalibriert mit Ergebnissen numerischer und physikalischer Modelle.* 3R-international, 47/1, 2008.

[7] I. E. Idelchik. *Handbook of hydraulic resistance.* Israel Program for Scientific Translations, Jerusalem, 1966.

[8] G. Bollrich. *Technische Hydromechanik, Band 1: Grundlagen.* Beuth, Berlin, 7th ed. 2013.

[9] U. Bierbaum, J. Hütter. *druckluft kompendium.* Weka Business Medien, Darmstadt, 6th ed. 2004.

[10] KAESER KOMPRESSOREN GmbH. *Berechnung des Druckabfalls.* Online: http://www.kaeser.at/Online_Services/Toolbox/Pressure_drop/default.asp (Aufruf 2015-04-15).

[11] W. Bohl, W. Elmendorf. *Technische Strömungslehre.* Vogel Business Media, Würzburg, 15th ed. 2014.

[12] M. Sielemann, F. Casella et al. *Robust Initialization of Differential-Algebraic Equations Using Homotopy.* Proc. 8th Int. Modelica Conference, Dresden, p. 75-85, 2011.

[13] F. Casella, M. Sielemann, L. Savoldelli. *Steady-state initialization of object-oriented thermo-fluid models by homotopy methods.* Proc. 8th Int. Modelica Conference, Dresden, p. 1-11, 2011.

[14] P. Junglas. *Praxis der Simulationstechnik.* Europa-Lehrmittel, Haan-Gruiten, 2014.