A Transient Data Extension to the GasLib

Dennis Gabriel[*], Richard Krug[†]

January 14, 2022

GasLib [1] provides a collection of realistic and artificial gas networks and corresponding load scenarios for the stationary case. In this document we describe an extension to this dataset containing transient scenarios.

This extension was developed within the collaborative research center TRR154 „Mathematical modeling, simulation and optimisation using the example of gas networks“. The goal was to provide a base set of data that can be used for computations and comparisons of different approaches. These datasets are available on the homepage of the research center (`https://www.trr154.fau.de/transient-data/`). The data is supplied in XML-format, as are the datasets of GasLib. In this document we explain the format and how the datasets have been constructed.

[*]Department of Mathematics, TU Darmstadt

[†]Department of Mathematics, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)

1 Description of the XML Files

In this section we take a closer look at the developed XML format. So far, we constructed two different kinds of files which we will describe by means of an example; for more details on XML see also [3]. The goal was to create a generalized format for many different kinds of files with different purposes. The base structure should always be the same, but one or the other information is added whiles others are left out. This idea led to the XML format we describe in the following.

To validate the constrcuted XML files we also developed XSD (XML Schema Definition) files. They can also be found on the research center's website (see `https://www.trr154.fau.de/transient-data/`). These schemes specify the format of the XML files and the types of the data; more information on XSD can be found in [4].

For an exemplary description we consider the Gaslib-11 network from the GasLib homepage (`http://gaslib.zib.de/`, [1]). For more information about the GasLib gas network instances and more general information see also the report on the GasLib libary [2]. For this network we supply the files `GasLib-11-sinus-InputData.bcd` and `GasLib-11-sinus_5000_60-initial.state`. The latter file contains a state for the Gaslib 11 network. More precisely, pressures and flows are given at the nodes and at the pipes, respectively. Furthermore, for pipes we also provide this data at spacial points accorcing to a given space discretization. For compressor stations and control valves, the pressure is replaced by the pressure difference and for valves then open/closed status is specified. The first file only contains information about nodes and not about edges. Instead, all nodes have an additional child node that specifies the timepoint to which the data belongs.

1.1 The `.state` file

We continue by taking a closer look on the `.state` file because it is more general than the `.bcd` file. The top node `initialdata` of the file has three child nodes. These are `metadata`, `nodes` and `edges`.

```
<metadata>
        <network>GasLib-11.net</network>
        <author>A05</author>
        <version>1.0</version>
        <date>2021-11-10</date>
        <speedOfSound value="340" unit="m_per_s"/>
        <pipemodel>ISO2</pipemodel>
</metadata>
```

Figure 1: An example for a `metadata` node in a `.state` file. It contains important general information on the file.

Figure 1 shows the node `metadata`. This node is characterized by a summary of important general information about the file and the setting of the dataset. Its first child node is `network`, which gives the name of the network file to which the `.state` file belongs. Furthermore, `metadata` contains the author, the version and the date. This information makes it possible to identify the file unambiguously. After that, a value and unit for the speed of sound is provided. This value can optionally be used for further calculations. The scheme also allows it to give a value for the friction coefficient if it is needed, but this is not given in the example. At last the pipe model that has been used is listed.

```
<node id="entry01" type="boundaryNode">
        <nodedata>
                <massflow value="15.223430916557934" unit="kg_per_s"/>
                <pressure value="53.0" unit="bar"/>
        </nodedata>
</node>
```

Figure 2: An example of a node in the section `nodes`. It has the child node `nodedata`, which itself has the child nodes `massflow` and `pressure`.

Going on, we consider the first sibling of the `metadata` node, that is the node `nodes`. Figure 2 shows an example. This section is a list of nodes from the network. Every node is described by its id and type. The id states the name of the specific node. The type it is `boundaryNode` or `innode`. If one wants to identify whether a node is source or sink, one has to take a look at the sign of the flow. In every node, information about pressure and flow is stored. More precisely, each node has a child node `nodedata` that itself contains two child nodes: `massflow` and `pressure`. Both nodes have attributes that provide the value and unit of the massflow and the pressure respectively.

```
<edge id="pipe08_N05_exit03" type="pipe">
        <edgedata>
                <space value="0.0" unit="m"/>
                <massflow value="22.677777777777777" unit="kg_per_s"/>
                <pressure value="46.12179630432075" unit="bar"/>
        </edgedata>
        <edgedata>
                <space value="5000.0" unit="m"/>
                <massflow value="22.677777777777777" unit="kg_per_s"/>
                <pressure value="45.89120822674979" unit="bar"/>
        </edgedata>
        ...
</edge>
```

Figure 3: An example of an edge of type `pipe` in the section `edges`. It has child nodes `edgedata` that themselves have child nodes `space`, `massflow` and `pressure`.

The last child node of `initialdata` is `edges`. This node cotains a list of edges similar to the section of `nodes`. Possible choices for child nodes are `pipes`, `valves` or `compressor stations`. Figure 3 shows an example for the described pipe structure. The structure of this child node is similar to the structure of the nodes from the `nodes` section. The top node `edge` has the attributes `id` and `type`. Below that we have `edgedata`. This yields a major difference in comparisson to the `nodes` section. If the edge type is pipe, edgedata has the child node space with attributes value and unit. It refers to a specific spatial point on the pipe according to a given (here equidistant) space discretization. Because of that, edges of type pipe have the same number of edgedata child nodes as the pipe has spatial points. Besides the information about the space, edgedata also contains child nodes massflow and pressure similar to nodes.

```
<edge id="V01_N01_N03" type="valve">
        <edgedata>
                <massflow value="0.0" unit="kg_per_s"/>
                <open value="0"/>
        </edgedata>
</edge>
```

Figure 4: An example of an edge of type pipe in the section edges. It has a child node edgedata that itself has child nodes massflow and open. The latter describes the staus of the valve.

The next possible choice for a type of an edge are the valves. An example can be found in figure 4. In comparison to the pipes, the valves child node valvedata has no child node for space and thus we do not need several instances. Furthermore, we have no node for pressure, but instead a node open, which describes the status of the valve. If it is open, the value is 1, if it is closed, the value is 0.

```
<edge id="CS01_entry03_N01" type="compressorStation">
        <edgedata>
                <massflow value="53.27911316234031" unit="kg_per_s"/>
                <pressureDifference value="5.144230483504607" unit="bar"/>
        </edgedata>
</edge>
```

Figure 5: An example of an edge of type compressorStation in the section edges. It has a child node edgedata that itself has child nodes massflow and pressureDifference.

The last possible choice are the compressor stations. Their structure is similar to the valves, but the pressure difference is provided instead of the status. It gives us the difference of pressure between the from and to node of the compressor station. An example is shown in Figure 5.

Note that there are actually more possible choices for edges, for examlpe control valves, resistors or shortpipes. These are established in a similar structure as pipes, valves and compressor stations, but we will not give an example here.

1.2 The .bcd file

The .state file provides an initial state for further work. For computations over time, we also need information about pressure and/or flow at certain timepoints. These specifications can be found in the .bcd (boundary conditions) file. For every node and every time point according to a given time discretization, values for flow and/or pressure are provided.

```xml
<metadata>
	<network>GasLib-11.net</network>
	<author>A05</author>
	<version>1.0</version>
	<date>2021-11-09</date>
	<timeInterval start="0" end="86400" unit="s"/>
	<speedOfSound value="340" unit="m_per_s"/>
</metadata>
```

Figure 6: An example for a `metadata` node in a `.bcd` file. The difference to a `metadata` node in a `.state` file is the node `timeInterval` with start- and endtime and their unit (here in seconds).

The structure of the `.bcd` file is similar to the structure of the `.state` file. The top level node `boundarydata` has two child nodes. These are `metadata` and `nodes`. The first is almost the same as in the `.state` file with one addition. Here `metadata` is complemented by an additional node `timeInterval`, which specifies the start and end of the considered time interval together with the unit of the time. Figure 6 shows an example for the `metadata` node in `GasLib-11-sinus-InputData.bcd`.

```xml
<node id="exit01" type="boundaryNode">
	<nodedata>
		<time value="0" unit="s"/>
		<massflow value="-28.34722222222222" unit="kg_per_s"/>
	</nodedata>
	<nodedata>
		<time value="60" unit="s"/>
		<massflow value="-28.359590992014123" unit="kg_per_s"/>
	</nodedata>
	<nodedata>
		<time value="120" unit="s"/>
		<massflow value="-28.37195952632247" unit="kg_per_s"/>
	</nodedata>
	...
</node>
```

Figure 7: An example for a node from the `nodes` section in a `.bcd` file. The difference to such a node in a `.state` file is the node `time` with a value and its unit (here in seconds). The value describes a certain timepoint that is considered at this datapoint. In this case the massflow is specified at every timepoint.

The second child node of `boundarydata`, `nodes` is a list of nodes in the corresponding network. Again, the structure is similar to the `nodes` section in the `.state` file with one important difference, which is shown in Figure 7. Here the node `node` has more then one child node `nodedata`. More precisely, it has one child node for every timepoint according to a given (here equidistant) time discretization. All `nodedata` nodes contain a node `time` with two attributes to specify a certain timepoint. These are the value and a unit. The value is a time on the time interval given in `metadata` together with its unit. An example for this structure is given in Figure 7. Here the massflow is specified at all timepoints. Another possibility is to specify the pressure.

```
<node id="entry02" type="boundaryNode">
        <nodedata>
                <time value="0" unit="s"/>
                <pressure value="51" unit="bar"/>
        </nodedata>
        <nodedata>
                <time value="86400" unit="s"/>
                <pressure value="51" unit="bar"/>
        </nodedata>
</node>
```

Figure 8: An example for a node from the `nodes` section in a `.bcd` file. The difference to such a node in a `.state` file is the node *time* with a value and its unit (here in seconds). The value describes a certain timepoint that is considered at this datapoint. In this case the pressure is specified at the start and end timepoint of the time interval.

In general, the files we provide have the pressure specified at the entries and the massflow specified at the exits. Figure 8 shows an entry with information about the pressure. Here we can also see a special case of our format: In this case, the pressure is constant over all timepoints. Instead of adding a child node for all timepoints with identical pressure value, there are two timepoints provided, one for the start and one for the end.

Note that this case where the pressure is specified at the entries and the massflow is specified at the exits is only an example. Our format also supports other configurations where for example flow is given at entries and exits. One can also mix it up and specify the flow at some entries and some exits. More precisely, in this format we only know if a boundary node is an exit or entry by the sign of the flow. If the flow changes through time an exit might become an exit, but this is also supported by our format.

2 Construction of datasets

In this section, we describe how the datasets that consist of a `.state` and a `.bcd` file each are constructed.

2.1 The `.bcd` file

The provided `.bcd` files contain pressure and massflow data at boundary nodes for a time horizon of 24h. The data values are inspired by the stationary nomination files from GasLib which divides boundary nodes into entry and exit nodes.

At the exit nodes, we give the massflow q as a vertical shifted sine curve with a period of 24h. The massflow at the initial time point $t = 0$ is taken from the GasLib nomination file that contains volumetric flow rates. Using the norm density in the GasLib network file, one can calculate the respective mass flow. The amplitude of the sine wave is set to 10% of the initial massflow. This yields the formula

$$q(t) = q(0) + 0.1 \sin\left(t \frac{2\pi}{24\text{h}}\right) q(0)$$

for the massflow at each exit node for $t \in [0, 24\text{h}]$. This massflow function is then sampled with a time difference of 60s.

The pressures at the entries are set to constant values which were chosen by hand.

2.2 The `.state` file

The provided `.state` files contain a stationary solution for the boundary values at the initial time point $t = 0$.

For the pipes we used the ISO4 model

$$\frac{\partial q}{\partial x} = 0, \tag{1a}$$

$$\frac{\partial p}{\partial x} = -\frac{\lambda c^2}{2DA^2} \frac{q|q|}{p} - \frac{gh'}{c^2} p \tag{1b}$$

where q and p denote the massflow and the pressure, c and g are the speed of sound and the gravitational acceleration, and A, D, and h' model the cross section area, the diameter, and the slope of the pipe. The friction factor λ is given by the law of Nikuradse

$$\lambda = \left(2 \log_{10}\left(\frac{D}{k}\right) + 1.138\right)^{-2}$$

where k denotes the roughness of the pipe.

Equation (1a) implies that each pipe has a single flow. By using implicit finite differences we discretize (1b) equidistantly as

$$\frac{p_{k+1} - p_k}{\Delta x} = -\frac{\lambda c^2}{2DA^2} \frac{q|q|}{p_{k+1}} - \frac{gh'}{c^2} p_{k+1}.$$

Now, the stationary gas network problem can be written as a finite dimensional non-linear optimization problem. Minimizing the sum of the pressure differences in the compressor station in the network then yields the data of the `.state` files.

References

[1] M. Schmidt, D. Aßmann, R. Burlacu, J. Humpola, I. Joormann, N. Kanelakis, T. Koch, D. Oucherif, M. E. Pfetsch, L. Schewe, R. Schwarz, and M. Sirvent. GasLib – A Library of Gas Network Instances. *Data*, 2(4):article 40, 2017.

[2] M. Schmidt, D. Aßmann, R. Burlacu, J. Humpola, I. Joormann, N. Kanelakis, T. Koch, D. Oucherif, M. E. Pfetsch, L. Schewe, R. Schwarz, and M. Sirvent. Gaslib—a library of gas network instances. Technical report, Optimization Online, 2017. `http://www.optimization-online.org/DB_HTML/2015/11/5216.html`.

[3] Extensible Markup Language (XML). `https://www.w3.org/XML/`.

[4] W3C XML Schema Definition Language (XSD). `https://www.w3.org/TR/xmlschema11-1/`.