ModSCA
MODular Simulator for Compressed Air

User Manual

Giusi Quartarone, Gianluca Sala, Norma Anglani

1 ModSCA: introduction

This user manual has been conceived to show how to use ModSCA: a new simulator for compressed air system (in the following, CAS) developed in LABAC at the Department of Industrial and Information Engineering, University of Pavia.

1.1 Overview on the efficiency of compressed air systems

The consumptions related to compressed air systems represent a relevant slice of the whole industrial electricity consumption. Nonetheless its diligent management still does not seem to be taken into sufficient consideration: only the 10% of the input energy is transferred into compressed air at the end user. Moreover, the research field shows scarce interest on CAS: limited is the number of works focused on how to reduce the energy consumption and few dedicated softwares have been designed and made available to end users.

The possibility of having a simulator is of paramount importance for assisting those companies whose annual operating costs of compressed air production can account for high share of the total electricity bill for specific users. The need of keeping under control how the system is working is fundamental from an energy savings point of view. Moreover, a superficial knowledge of the operation, design and evaluation of energy requirements and critical elements of a compressed air system, has highlighted the need to provide this information.

2 Simulator

2.1 Software's structure

ModSCA needs a simulator environment, this version is implemented for Matlab R2012b or following releases.
The folder `simulator` contains all the files necessary for a correct use of ModSCA. The folder contains the following item:

- `devices`; this folder contains files `.slx`: these are the modules emulating the main devices installed in a compressed air system. The functioning of each device is modelled inside a **library-block**:
  - air_dryer.slx
  - circular_network.slx
  - end_user.slx
  - filter.slx

  - load_unload_compressor.slx
  - pipe.slx
  - pressure_reducer.slx
  - receiver.slx
  - system_parameters.slx
  - vsd_PI_compressor.slx
  - vsd_MPC_compressor.slx

- `subsystems`; this folder contains five sub folders (`air_dryer`, `compressors`, `constants`, `parameter`, `pipe`. For more information, see the **Appendix A**) containing the blocks used to build the functionality of the library blocks.

- `parameters.m`; this file contains all the parameters will be used to perform the simulations. If some parameters reported in the file are not stored in the workspace, the simulation will not run correctly: an error will show you during the simulation running. The file suggests default values can be used if they are unknown; moreover, the files contains the units characterizing each value and some comments to explain what the parameter does. For more details, see **Appendix B**.

2.2 Library blocks

Each block can be easily recognized through an image of the device is emulating, and through the input and output signals. The Figure 1, represent the block modeling the a load/unload compressor: the block not receive any signal value, and provide, as output, the power and energy consumption and the flow provided by the compressor. To execute a simulation, the input and output values must to be connect to the proper signals.

With a double click, it will open a mask containing the parameters required during the simulation. The parameters' value can be set manually, introducing the real value, or modifying the corresponding value stored in the file `parameters.m`. To help the user, the mask shows a brief description of the parameters and their measurement units.

With the right mouse button, you can click `Look Under Mask`: the blocks and sub-blocks describing the mechanical, electrical and physical models will appear.

2.3 How to modify the model of each block

ModSCA allows the user to modify the mathematical models implemented inside each block-device: in this way it is possible to adjust the theoretical model to the real one.

To modify the mathematical model of the block

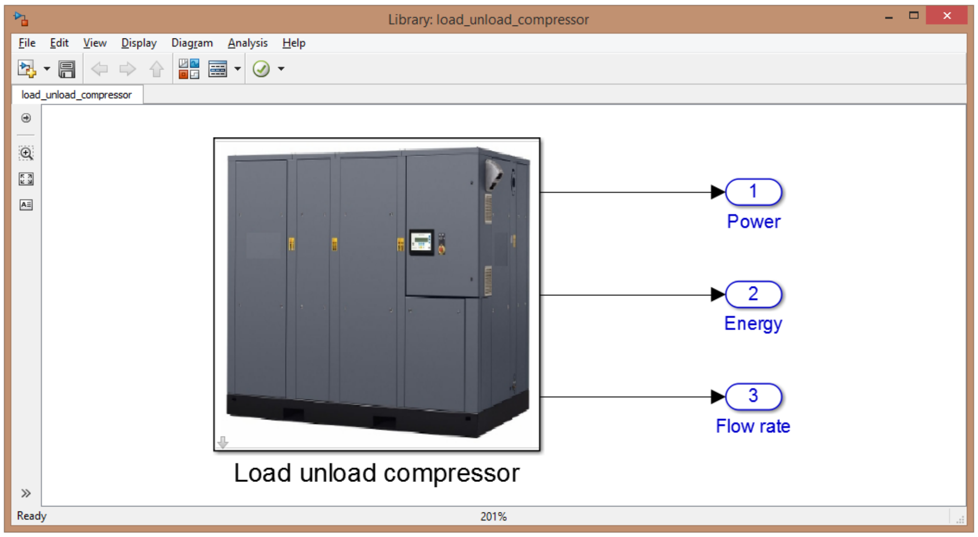

Figure 1: load unload compressor module

`Edit > Unlock Library`.
`Edit > Look Under Mask`.

3 Moduls

ModSCA is a modular simulator, so each element of a compressed air system is represented by a block whose internal model is not visible to the users, but modifiable by them. In this way, the user can reproduce his compressed air system or design a new one, searching the devices from the simulator library.

The modules already implemented emulates the mainly devices installed into the LABAC: load/unload compressor, adjustable speed drive compressor, receiver, air dryer, air filter, network, pressure reducer, flow profile required by the end-users and system.

3.1 Compressors

ModSCA can simulate two different kind of compressors: a load/unload compressor and an adjustable speed drive compressor. Through the proper parameters setting it is possible to simulate compressors of different sizes and working point.

3.2 Load/Unload compressor

This module simulates a load/unload compressor: it runs at full load as long as the outlet pressure reaches the upper activation pressure setting (`pmax` parameter), which causes the compressor goes to unload. When unloaded, the compressor does not deliver air but the electric motor keeps on running, consuming from the grid roughly $20-30\%$ of the power (this value is stored in `fraction_power_unload`), required when it is in its load mode. If the compressor keeps standing in its unload mode for several seconds (set in `t_on_off` parameter) and if maximum number of switches-off per hour (`n_avviamenti` parameter) has not been already achieved, then the compressor switches-off. When the pressure reaches the load value (in `pmin` parameter), the compressor switches in unload mode for a period whose value can be set in the parameter `time_from_off_to_on`. After this period, the compressor switches in load mode.

The power absorbed by the compressor in load mode is reported in the following:

$$P = P_c + P_{oil\,pump} + P_{fun} \tag{1}$$

where $P_{oil\,pump}$ is the power absorbed by the pump that moves the lubricant oil (the value is reported in `oil_pump_power`), P_{fun} is the power absorbed by the fun for the air cooling (`fan_power`) and P_c is calculated as follows:

$$P_c = \frac{p_{amb}\dot{V}}{10^3\eta_{pol}\eta_m\eta_{gear}}\frac{n}{n-1}\Big[\big(\frac{p_{out}}{p_{amb}}\big)^{\dfrac{n-1}{n}} - 1\Big] \quad [kW] \tag{2}$$

where p_{amb} is the ambient pressure set in `pressure_amb`, $\dot{V}$ is the flow provided by the compressor evaluated at FAD conditions (this value is stored in `fad`). η_{pol}, η_m and η_{gear} represent respectively the polytropic compression efficiency, the motor and gear efficiencies. To set these values, you have to refer to `rendimento_politropica`, `rendimento_motore` and `rendimento_trasmissione`. p_{out} is the relative pressure of the compressed air produced.

The parameter n represent the polytropic coefficient and is calculated by ModSCA through the Equation 26.

3.2.1 Input/Output signals

Input signal: Although is not visible, the module receive as input signal the pressure inside the receiver. This signal is not show because it isn't a real input for this module but a required value to evaluate other signals.

Output signal:

- power: instantaneous power absorbed by the compressor in kW.
- energy: energy consumed by the compressor kWh.
- flow rate: it represents the instantaneous flow in m^3/s provided by the compressor in FAD conditions.

3.2.2 Main sub-blocks

Inside the folder `simulator/subsystems/compressors` there are the sub-block used to implement the compressor's functionalities.

- `compressor_control_lnl.slx`: through the knowledge of the system pressure, this sub-block identify the compressor state: load, unload or off.
- `compressor_power_lnl.slx`: it contains the implementation of the Equation 2.
- `power_dynamics_from_loadToUnload.slx`: it implements the dynamic of the power absorbed from the grid when the compressor switches from load to unload

  $$P_{unload} = P * (1 - fp)e^{-\frac{(t-t^*)}{v}} + P * fp \tag{3}$$

  where fp represent the value contained in `fraction_power_unload` and v represent the velocity from load to unload whose value is stored in `velocity_loadTOunload`.

3.2.3 Data stored in the Matlab's workspace

- `compressor_state`: it contains the state of the compressor for each time sample during the simulation. It can assume the value 1, 2 or 3: they represent respectively the off, unload and load state.
- `power`: it represents the instantaneous power measured in $[kW]$.
- `compressor_flow_rate`: it represents the instantaneous compressor flow in $[m^3/s]$ at FAD conditions.
- `energy`: it represents the total energy consumed by the compressor from the beginning of the simulation until the present time. It is evaluated in kWh.
- `load_time`: it represents the time in which the compressor worked in load mode. It is evaluated in s.
- `unload_time`: it represents the time in which the compressor worked in unload mode. It is evaluated in s.
- `stop_time`: it represents the time in which the compressor was switched-off. It is evaluated in s.

3.3 Adjustable speed compressor controlled with PI

This module simulates a variable speed drive compressor: it runs modulating the motor speed, hence the output flow, as long as the outlet pressure reaches an upper pressure setting (`p_off_vsd` parameter), which causes the compressor goes in off mode; the motor starts to decrease its velocity: each second, the motor speed is reduce of `max_delta_speed`. Depending on the motor speed before the stop command, the compressor take some seconds to completely stop the motor. When the pressure reaches the load value (in `p_load_vsd` parameter), the compressor switches in modulating mode: the velocity start to increase with the same velocity.

When loaded, the compressor modulates its velocity between a bounded range (respectively in `min_speed` and `max_speed`) depending on the pressure inside the system. Generally, the motor speed is controlled through a PI control scheme: the control law implemented is the following

$$\Delta n_t = min\Big\{\Delta n_{max}, K\big(k_p \Delta e_t + k_P k_I e_t ts\big)\Big\} \tag{4}$$

where Δn_{max} is the maximum motor speed variation allowed every second and stored in `max_delta_speed` , k_P and k_I are respectively the proportional and integral gain (they are stored in `kp` and `ki`), K is a scaling factor (in `scaling_factor`) e_t is the difference between the pressure set point (in `pressure_setpoint`) and the system pressure at time t, while $\Delta e_t = e_t - e_{t-1}$.

After computing Δn_t, it is possible to evaluate the motor speed at time t as follows:

$$\begin{cases} n(t) = n(t-1) + \Delta n_t & \text{if } n_{min} < n(t) < n_{max} \\ n(t) = n_{min} & \text{if } n(t) \leq n_{min} \\ n(t) = n_{max} & \text{if } n(t) \geq n_{max} \end{cases} \tag{5}$$

Depending to the motor speed, the flow provided by the compressor follows the dynamic:

$$q(t) = \frac{q_{max} - q_{min}}{n_{max} - n_{min}}(n(t) - n_{min}) + q_{min} \tag{6}$$

where q_{max} is the maximum flow can be produced by the adjustable speed compressor (stored in `max_flow`), while q_{min} (`min_flow`) is the minimum flow can be provide the compressor.

The absorbed compressor power, depending on the flow is providing the compressor, was modelled as follows:

$$P = (a_1 + a_2 p_{op}) + q(a_3 + a_4 p_{op}) \tag{7}$$

where p_{op} is the operative pressure of the compressor, q is the flow provided by the compressor, while $a_1,\ a_2,\ a_3,\ a_4$ are parameters describing the curve $P(q,p)$: they are generally provided by the compressor's manufacturer.

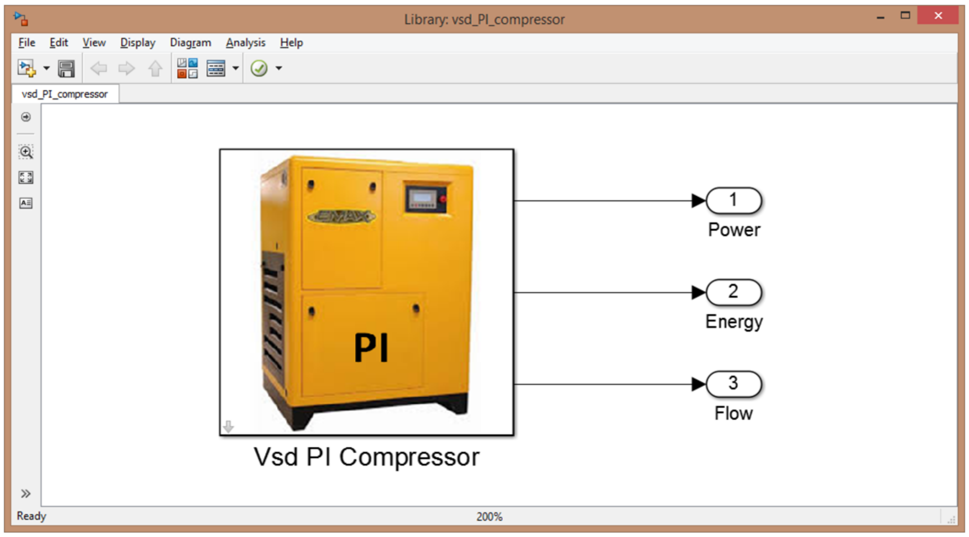
Library: vsd_PI_compressor
File Edit View Display Diagram Analysis Help
vsd_PI_compressor
PI
1
Power
2
Energy
3
Flow
Vsd PI Compressor
Ready
200%

Figure 2: Vsd PI compressor module

3.3.1 Input/Output Signals

Input signal: Although is not visible, the module receive as input signal the pressure inside the receiver. This signal is not show because it isn't a real input for this module but a required value to evaluate other signals.

Output signal:

- power: instantaneous power absorbed by the compressor in kW.
- energy: energy consumed by the compressor kWh.
- flow: it represents the instantaneous flow in m^3/s provided by the compressor in FAD conditions.

3.3.2 Main sub-blocks

Inside the folder `simulator/subsystems/compressors` there are the following sub-blocks:

- `compressor_control_vsd_throttle.slx`: depending on the system pressure and the pressure settings, it evaluate the compressor state (the number 3 represents the on mode, while 1 the off mode).
- `PI_control.slx`: it implements the PI control to modulate the motor speed (Equations (4) and (5)).
- `reduction_motor_speed.mdl`: it implements the motor speed variation when the compressor switch from load to off and vice versa.

- `throttle_vsd_flow.slx`: it calculates the flow provided by the vsd (Equation (6), depending on the instantaneous motor speed.
- `throttle_vsd_power.slx`: depending on the flow is providing the compressor and the operative pressure of the compressor, the sub-block implements the Equation (7).

3.3.3 Data stored in the Matlab's workspace

- `power`: it represents the instantaneous power measured in kW.
- `compressor_flow_rate`: it represents the instantaneous compressor flow in $m3/s$ at FAD conditions.
- `energy`: it represents the total energy consumed by the compressor from the beginning of the simulation until the present time. It is evaluated in kWh.
- `load_time_vsd`: it represents the time in which the compressor worked in load mode. It is evaluated in s.
- `stop_time_vsd`: it represents the time in which the compressor was switched-off. It is evaluated in s.

3.4 Adjustable speed compressor controlled with MPC

The proposed method is based on model predictive control that is an optimal controller based on the future air demand forecasting. Differently from the PI controller, the MPC does not required the setting of a set-point but only of a control range in which the pressure can vary. As a consequence of the reduction of the pressure levels, a smaller amount of energy is absorbed by the compressor and a reduction of air leaks occurs. It is possible because the MPC has the ability to anticipate future events, using this information to take control actions, accordingly.
For a comprehensive implementation of this control scheme, forecasting techniques need to be applied to assess end-users air requirements. Model predictive control is an optimal-control based method to select control inputs by minimizing an objective function defined in terms of both present and predicted system variables.
MPC is also known as Receding Horizon Control because at the k-th time it solves an optimal control problem over a finite future horizon of t $\in \{0, 1, 2, . . . ,N - 1\}$ steps, then applying only the first optimal move. N is generally called receding horizon (parameter `N`).
At each k-th instant, the horizon is displaced towards the future.
The MPC strategy implemented in this model aims to minimize the power consumption P_{AS}(k) along the prediction horizon by optimizing the motor speed. The model of the power consumption is reported in Equation 7 .
First, we introduce the constraints of a typical screw compressor. Due to electrical and mechanical reasons, the motor speed can vary between two thresholds.
Moreover, the motor speed variation between two consecutive time instants is also bounded between $[\Delta n_{min}\ ;\ \Delta n_{max}]$. Therefore, the following constraints must be guarantee for a correct motor functioning.

$$\begin{cases} n_{min} \leq n(k) \leq n_{max} & k \geq 0 \\ \Delta n_{min} \leq n(k) \leq \Delta n_{max} & k \geq 0 \end{cases} \tag{8}$$

where: $\Delta n(k) = n(k) - n(k-1)$ (parameter `max_delta_speed`) and $n(-1) = 0$.
Moreover, because the flow provided by an adjustable speed compressor can varies between two thresholds, the following constraint must be fulfilled.

$$q_{min} \leq q(k) \leq q_{max} \tag{9}$$

where q_{max} is the maximum flow can be produced by the adjustable speed compressor (stored in `max_flow`), while q_{min} (`min_flow`) is the minimum flow can be provide the compressor. Depending to the motor speed, the flow provided by the compressor follows the Equation 6.

Similarly, the system pressure p(k) can vary between two thresholds:

$$p_{min} \leq p(k) \leq p_{max} \tag{10}$$

where p_{min} and p_{max} are respectively the minimum (parameter `pmin`) and the maximum (parameter `pmax`) pressure levels allowed in the system.

The system pressure p(k), evaluated at time instant k, can be written as:

$$p(k) = p(k-1) + \Delta p(k-1) \tag{11}$$

Using the ideal gas law, the Equation 11 can be written as reported below:

$$p(k+1) = p(k) + \frac{q_c(k) - q_{usr}(k)}{V}\, p_0 \tag{12}$$

where q_c is the instantaneous flow rate provided by the compressor room (this value is the FAD provided by the compressor and stored in `fad` parameter), q_{usr} is the instantaneous flow rate required by the users, (this value is the volume flow rate required by the end-user, expressed in FAD condition and stored in `flow_out` parameter). V represents the receiver volume reported in `volume_tank`. p_0 is the pressure of the FAD condition.

So, at all time instants k, we solve the following linear optimization problem:

$$F(n(k)) = \min_{n(0:N-1|k)} \sum_{t=0}^{N-1} P_{AS}(n(t|k)) \tag{13}$$

In order to solve this optimization problem, the script `"MPC.m"` it's needed. This script contains the formulation of the problem written using YALMIP [1], a language for advanced modelling and solution of convex and non-convex optimization problems, implemented as a free toolbox for MATLAB. So you have to add to the Matlab path `"MPC.m"` and `"YALMIP"` folder both.

3.4.1 Input/Output Signals

Input signal: Although is not visible, the module receive as input signal the pressure inside the receiver. This signal is not show because it isn't a real input for this module but a required value to evaluate other signals.

Output signal:

- power: instantaneous power absorbed by the compressor in kW.

[1]http://users.isy.liu.se/johanl/yalmip/pmwiki.php?n=Main.HomePage

- energy: energy consumed by the compressor kWh.
- flow: it represents the instantaneous flow in m^3/s provided by the compressor in FAD conditions.

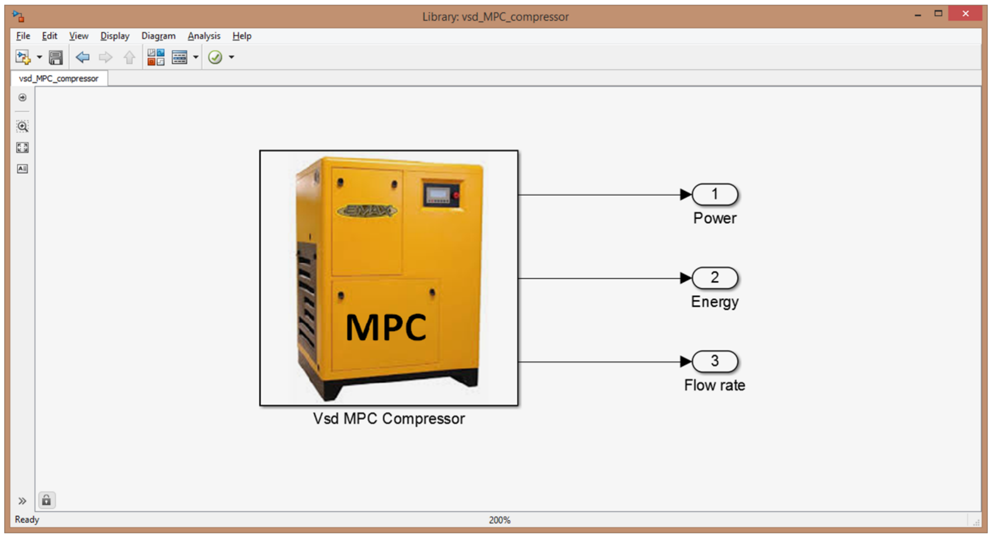
Library: vsd_MPC_compressor
File Edit View Display Diagram Analysis Help
vsd_MPC_compressor
MPC
Vsd MPC Compressor
1
Power
2
Energy
3
Flow rate
Ready
200%

Figure 3: Vsd MPC compressor module

3.4.2 Main sub-blocks

Inside the folder `simulator/subsystems/compressors` there are the following sub-blocks:

- `throttle_vsd_flow.slx`: it calculates the flow provided by the vsd (Equation (6), depending on the instantaneous motor speed.
- `throttle_vsd_power.slx`: depending on the flow is providing the compressor and the operative pressure of the compressor, the sub-block implements the Equation (7).

3.4.3 Data stored in the Matlab's workspace

- `power`: it represents the instantaneous power measured in kW.
- `compressor_flow_rate`: it represents the instantaneous compressor flow in $m3/3$ at FAD conditions.
- `energy`: it represents the total energy consumed by the compressor from the beginning of the simulation until the present time. It is evaluated in kWh.

3.5 Receiver

A compressed air network is made up of one or more receivers and the pipelines. The vessels, usually placed next to the compressors, decouple the compressor production from the end user demand avoiding frequently switch from load to unload mode. If it happens, it underlies an incorrect sizing scheme. Thus, a proper storage sizing becomes very important from an energy saving standpoint.

The module implements a pressure level variation, depending on air flowing in and out the receiver.

$$p(t+1) = p(t) + \frac{q_c - q_{usr} - q_{leaks}}{V}\, p_0 \tag{14}$$

where q_c is the instantaneous flow rate provided by the compressor room (this value is the FAD provided by the compressor and stored in `fad` parameter), q_{usr} is the instantaneous flow rate required by the users, (this value is the volume flow rate required by the end-user, expressed in FAD condition[2] and stored in `flow_out` parameter), and q_{leaks} is the leaks' flow whose value is calculated from the sub-block `FAD_loss.slx` . V represents the receiver volume reported in `volume_tank`. p_0 is the pressure of the FAD condition.

3.5.1 Input/Output signals

Input signal: Compressor's flow rate $[m^3/s]$ expressed in FAD condition. Moreover, although is not visible, the module receive another input signal: the end-user flow. This signal is not show because it isn't a real input signal but a value required to evaluate other signals.

Output signal: Gauge pressure inside the system $[Pa_g]$.

3.5.2 Main sub-blocks

Inside the folder `simulator/subsystems/tank` there is the sub-block `FAD_loss.slx` that calculates the total mass loss of compressed air along the distribution system and converts it in volume flow rate expressed in FAD condition $[m^3/s]$.

3.5.3 Data stored in the Matlab's workspace

- `tank_pressure`: for each sample time, it contains pressure into the receiver in $[Pa]$ gauge.
- `FAD_loss`: for each sample time, it contains the volume flow rate lost along the distribution system.

[2]FAD condition: $p_0 = 1\ bar$ and $T_0 = 20°C$

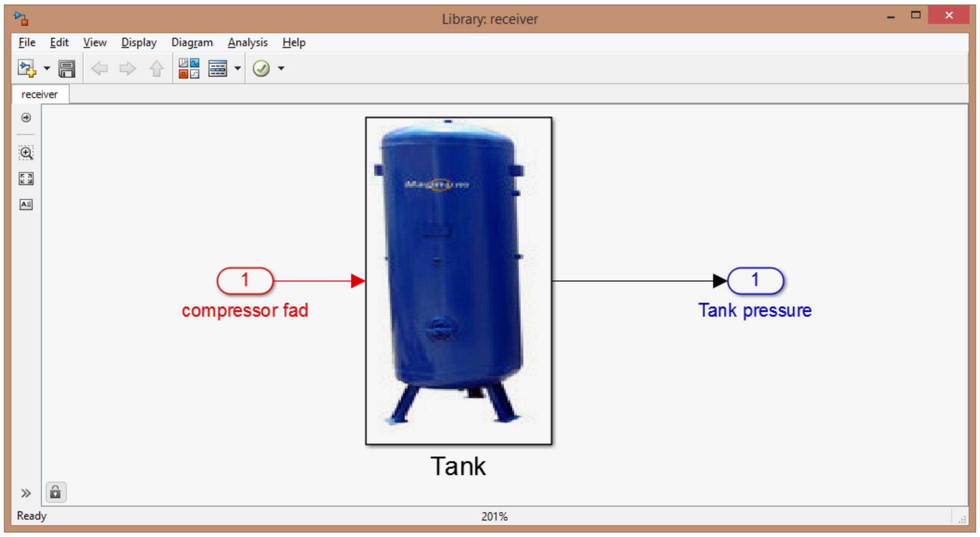

Figure 4: Receiver module

3.6 Air dryer

The air dryer removes water vapour from compressed air: excessive water in compressed air, either in liquid or vapour phase, can cause a variety of operational problems for users of compressed air. These include freezing of outdoor air lines, corrosion in piping and equipment, malfunctioning of pneumatic process control instruments, fouling of processes and products, and more.

Exist several type of air dryer: regenerative desiccant dryers, often called ”regens” or ”twin tower” dryers, refrigerated dryers, deliquescent dryers and membrane dryers.

The device modelled in ModSCA is a refrigerated air dryer: it is refrigeration system that reduces the compressed air temperature, hence the limit of water vapour saturation (dew point); lowering the air temperature, the water starts to condensate. Smaller is the output air temperature, the smaller is the air moisture content.

Knowing the temperature of the air sucked in by the compressor and the air relative humidity (this value is stored inside `U_rel`), it is possible to calculate the amount of water vapour $[g/m^3]$ contained inside the air: obviously, this is the same quantity of water vapour after the compression process. The amount of water vapour, in saturated conditions, comes from ¨*Handbook of thermodynamics tables and charts*, K. R. Njevi¨: they have been reported in ModSCA inside a lookup table. When the air flows inside the air dryer, its temperature decreases: knowing the reduced outlet air temperature and the air pressure it is possible to find, through tabulated values[3], the amount of water vapour contained in the air at that conditions; the difference between the amount of water sucked in by

[3]The values used in ModSCA comes from the ¨*Bigino dell'aria compressa*,2000¨

the compressor and the water inside the air flowing out the air dryer represents the ideal amount of water inside the refrigerant. The real amount of water depends on the air dryer efficiency (in `rendimento_air_dryer`): the final water vapour inside the compressed air after the refrigeration process is the difference between the starting value and the real water vapour inside the refrigerant.

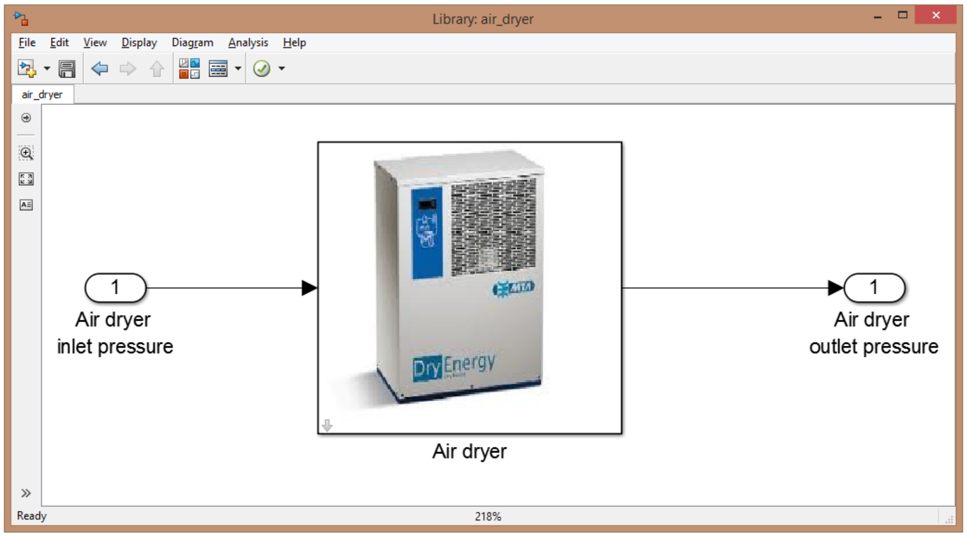

Figure 5: Air dryer module

3.6.1 Input/Output signals

Input signal: air pressure $[Pa_g]$ upstream the air dryer.

Output signal: air pressure $[Pa_g]$ downstream the air dryer

3.6.2 Main sub-blocks

Inside the folder `simulator/subsystems/air_dryer` there is the sub-block `saturated_vapour.slx`

3.6.3 Data stored in the Matlab's workspace

- `pressure_after_dryer`: for each sample time, it contains pressure after the air dryer in $[Pa_g]$.
- `water_vapour`: for each sample time, it contains the amount of water vapour $[g/m^3]$ remaining in the compressed air.

3.7 Filter

The compressed air filters are used to remove particle, dust and oil from compressed air. The air treatment devices are necessary to provide a proper quality of compressed air and to avoid an early devices' wear.

The air, passing through the filter, is subject to a pressure drop, increasing with the accumulated dirt in the filter. Even in the filter constantly cleaned, a pressure drop of at least 300 Pa is expected.

In Figure 6 the filter module in reported.

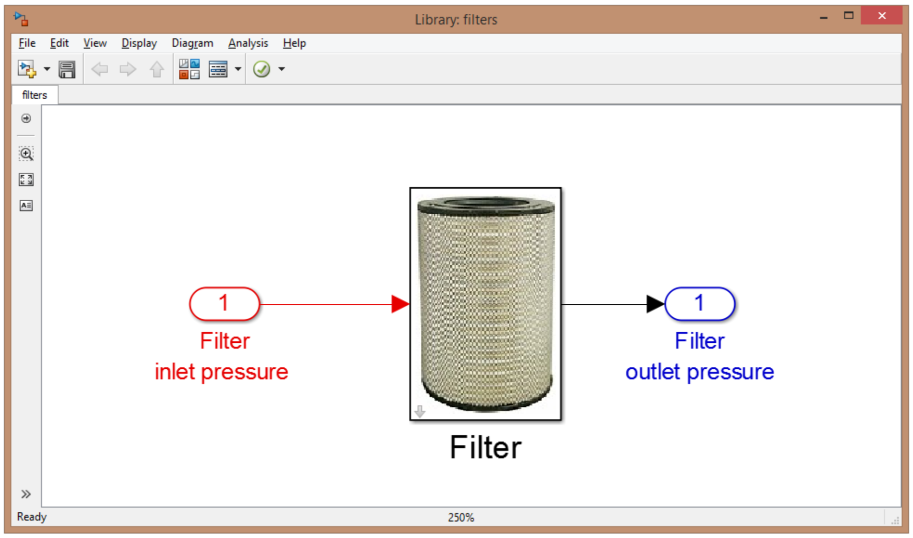

Figure 6: Air filter module.

The air filter module implements a pressure drop whose quantity is reported in `filter_pressure_drop`, hence

$$p_{out} = (p_{in} - \Delta p_f) \quad [Pa_g] \tag{15}$$

where Δp_f is the pressure drop that occurs inside the filter (`filter_pressure_drop`).

3.7.1 Input/Output signals

Input signal: Filter inlet pressure $[Pa_g]$.

Output signal: Filter Outlet pressure $[Pa_g]$.

3.7.2 Main sub-blocks

Inside the folder `simulator/subsystems/parameter` there is the sub-block `filter_p_drop.slx`

3.8 Radial Network

Inside a compressed air network, the air is subjected to a pressure drop due to mass losses, concentrated and dynamic losses. The first one depends on leaks can occur along the network, the second one is due to curves, junction etc. in the pipes, while the last one are due to frictions when the air flows inside the pipe.

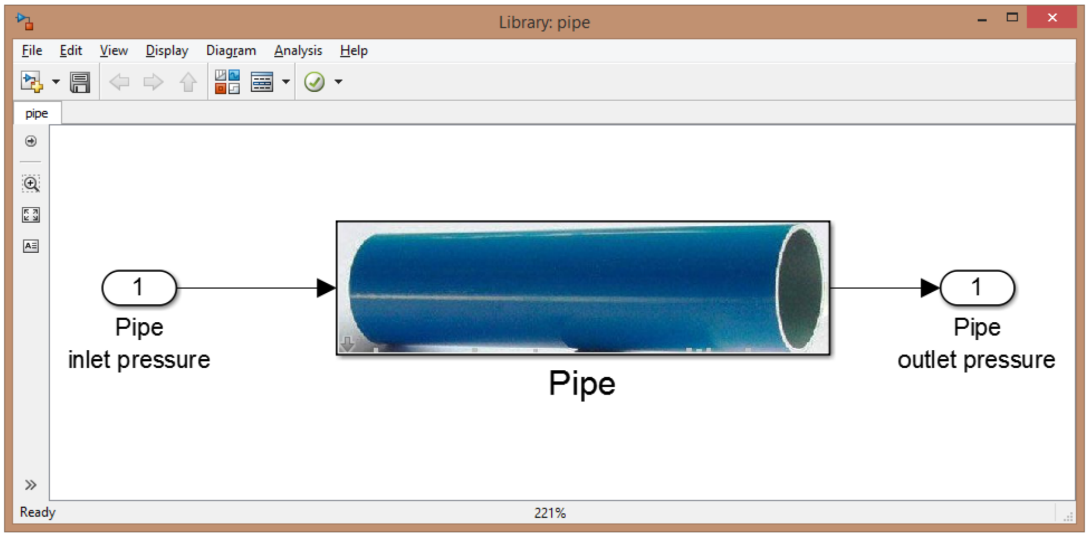

Figure 7: Radial distribution system module.

The dynamic losses are modeled in ModSCA through the Darcy's formula that can be used for incompressible fluids: we all know that the air is a compressible fluid but it is possible to demonstrate that if the Much number is smaller than 0.3, the air can be considered as an incompressible fluid. In fluid mechanics, the Mach number is a dimensionless quantity representing the ratio of speed of an object moving through a fluid and the local speed of sound. In the case considered this parameter can be calculate as follows:

$$Ma = \frac{v}{c} \tag{16}$$

where v is the air velocity inside the pipe while c is the speed of sound.

The Darcy's formula is presented below:

$$\Delta p = f\frac{\rho L v^2}{2D} \tag{17}$$

where f is the pipe's friction factor, whose value is calculated inside a sub-block will be shown in the following, and ρ is the air density evaluated as reported in Equation (29). D is the pipe's diameter, v the air velocity calculated as reported in Equation (18), L the equivalent length of the pipe: this value is the sum of the real length of the pipe, stored in `pipe_length`, and equivalent length of the curved, valves, gate valves etc. Inside these elements concentrated losses occur: through some tabulated value it is possible to describe the pressure drop inside each elements using an equivalent pipe length inside which occur the same pressure drop.

The values of the equivalent length used in ModSCA come from the Ingersoll-Rand Company (1980) and assume the values reported in Table 1, or their interpolation.

Table 1: Equivalent length (Ingersoll-Rand Company (1980))

Pipe Diameter in mm	25	40	50	80	100	125	150
Gate valve and curve	0.3 m	0.5 m	0.6 m	1 m	1.3 m	1.6 m	1.9 m
T curve and 90° curve	1.5 m	2.4 m	3.0 m	4.8 m	6 m	7.5 m	9.0 m
Gate valve (half opened)	5 m	8 m	10 m	16 m	20 m	25 m	30 m

3.8.1 Input/Output signals

Input signal: air pressure in $[Pa_g]$ evaluated at the beginning of the pipe. Moreover, although is not visible, the module receive another input signal: the flow that passes inside the pipe. This signal is not show because it isn't an input signal but a value required to evaluate other signals.

Output signal: air pressure in $[Pa_g]$ evaluated at the end of the pipe.

3.8.2 Main sub-blocks

Inside the folder `simulator/subsystems/pipe` there are the following sub-blocks `.slx`

- `air_velocity.slx`: this block calculate the speed of the air inside the pipe as follows:

$$v = \frac{\dot{m}_{usr}}{(D^2/4)\pi\rho} \tag{18}$$

- `darcy.slx` it implements the Equation (17).

- `iterazione_colebrook.slx`: this sub-block calculates the friction factor than assumes different values as shown in the Moody diagram reported in Figure (8).

  The diagram can be divided into two parts: the laminar and turbulent flow. We assume a laminar flow when the Reynolds number is smaller than 4000. In this case the friction factor can be calculated as follows:

$$f = \frac{Re}{64} \tag{19}$$

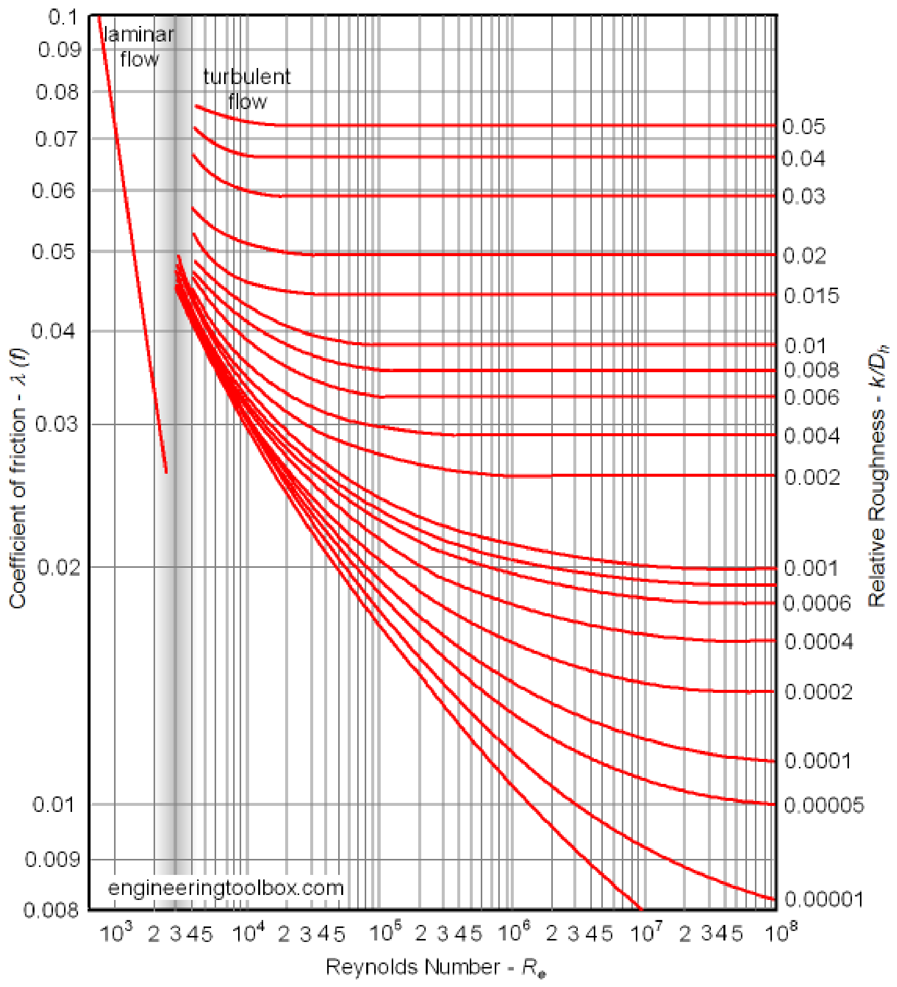

Figure 8: Moody diagram.

If the flow is turbulent, the friction factor is calculated considering the empiric equation of Colebrook-White, reported as follows:

$$\frac{1}{\sqrt{f}} = -2\log\left[\frac{\epsilon/D}{3.7} + \frac{2.51}{Re\sqrt{f}}\right] \tag{20}$$

where ϵ is the absolute roughness whose values are reported in Table (2), ϵ/D the relative one and Re is the Reynolds number calculated as Equation (28). The Equation (20) is transcendental meaning that it is necessary to use numerical methods to find the solution. We decide to use the iterative procedure proposed in "Clamond Didier. *Efficient resolution of the Colebrook equation*, 2009" and following reported.

Table 2: Absolute roughness

Material	Absolute roughness
Copper and aluminium	$0.001 < \epsilon < 0.003$
Plastic materials	$0.002 < \epsilon < 0.007$
Galvanized steel	$0.020 < \epsilon < 0.030$
Steel	$0.040 < \epsilon < 0.090$
Corroded steel	$0.200 < \epsilon < 1.000$

```
X1=k·Re·0,123968186335417556

X2=log(Re)-0,779397488455682028
F=X2-0,2
E1=(log(X1+F)+F-X2)/(1+X1+F)
F1=F-(1+X1+F+0,5·E1)·E1·(X1+F)/(1+X1+F+E1·(1+E1/3)
E2=(log(X1+F1)+F1-X2)/(1+X1+F1)
F2=F1-(1+X1+F1+0,5·E2)·E1·(X1+F1)/(1+X1+F1+E2·(1+E2/3)
F3=1,251292546497022842/F2
lambda=F3·F3
```

where $k = \epsilon/D$ is the relative roughness, Re is the Reynolds number.

- `pipe_equivalent_length.slx` it calculates the equivalent network length from Table (1).

3.8.3 Data stored in the Matlab's workspace

- `pressure_after_pipe`: the vector contains the sample of the air pressure $[Pa_g]$ after the pipe.

3.9 Circular Network

Independently of the number of installed devices and their placement, two basic layouts for the distribution system exist. A radial network is composed by a single line from the supply side to the points of usage. On the contrary, in a ring distribution system the compressed air flows in a closed loop header. Two pipelines link the supply side to the point of use, meaning that the total air hands out over two branches. Being smaller the air mass flowing inside each line, the air velocity is reduced, entailing in a smaller friction against the pipe walls and reduced pressure drops.

Unfortunately, the structure of simulink doesn't allow to use the same circular network block for system with different number of loads. So there were created 4 different blocks: the first has only one load `circular_network1.slx`, the second has two loads `circular_network2.slx`, the third has three loads `circular_network3.slx` and the fourth has four loads `circular_network4.slx`.

Furthermore, the blocks representing the end users had to be incorporated within the circular network block in order to avoid ambiguous references in the system.

If the number of loads is equal to n, the network can be divided in n+1 parts where n+1 different flows that are our unknowns pass. Since there are n+1 unknowns, n+1 equations will occur. These equations are: one equation that describes the pressure balance and n equations that represent the mass balances. In order to calculate the pressure drop in the circular network, the Darcy's formula (Equation 17) is implemented. There are some terms that can be simplified, so it can be used the first equation of the system 21.

$$\begin{cases} f_1 L_1 v_1^2 + ... + f_n L_n v_n^2 = f_{n+1} L_{n+1} v_{n+1}^2 \\ \dot{m}_A = \dot{m}_1 + \dot{m}_2 \\ \dot{m}_B = \dot{m}_2 + \dot{m}_3 \\ ...... \\ \dot{m}_N = \dot{m}_n + \dot{m}_{n+1} \end{cases} \tag{21}$$

For the computation of the friction factor it is used the same procedure seen in the radial network module. The sub-block `iterazione_colebrooke.slx` is incorporated within a Matlab Embedded function that solves the system 21.

3.9.1 Input/Output signals

Input signal: Inlet pressure of the network $[Pa_g]$. Moreover, although is not visible, the module receive another input signal: the end-user flow. This signal is not show but it's a value required to evaluate the other signals.

Output signal: there's no output signal.

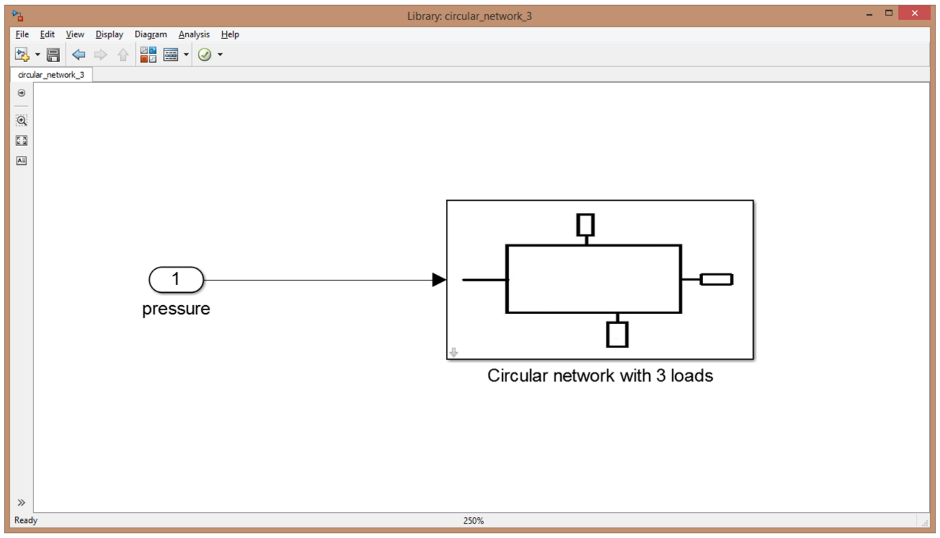
Library: circular_network_3
File Edit View Display Diagram Analysis Help
circular_network_3
1
pressure
Circular network with 3 loads
Ready
250%

Figure 9: circular distribution system with 3 loads.

3.9.2 Data stored in the Matlab's workspace

- `pressure_load`: the vector contains the sample of the air pressure $[Pa_g]$ for each load .
- `volume_flow`: the vector contains the sample of the volume flow rate $[m^3/s]$ expressed in FAD condition for each load .

3.10 Pressure reducer

Higher is the air pressure inside the system, bigger is the power absorbed by the compressor: for this reason, the pressure inside the system has to be as lower as possible. Most systems have one or more critical applications that determine the minimum acceptable pressure in the system. Generally the end-users operative pressures are different: in this cases, more than one compressed air system could be necessary; each of them has an own compressor room and network, resulting in a higher installation and maintenance costs. To reduce these costs, pressure reducer are generally used; pressure drops in the order of 40% of the compressor-discharged pressure are not uncommon.

In Figure (10) is reported the pressure reducer module: it emulates the pressure drop, providing, at the same time, the energy wasted to increase the air pressure to levels higher than required. The equation used to model the energy waste comes from ¨D. Kaya, P. Phelan, D. Chau, H. Sarac, *Energy conservation in compressed air systems*, 2002¨.

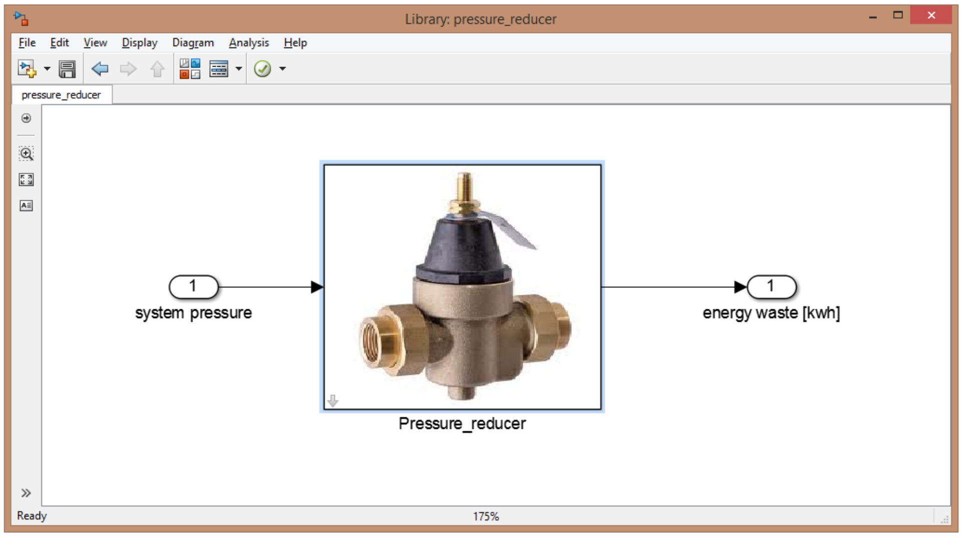

Figure 10: Pressure reducer module.

The total energy lost US in $[kWh]$ from the beginning of the simulation can be estimated as follows:

$$US = \sum_{t=1}^{T} (1.0 - FR_t) * P * ts \tag{22}$$

where FR_t is the ratio of the proposed power consumption to current power consumption based on maximum operating pressure: it is an a dimensional value. P is the power absorbed by the compressors while loaded (in $[kW]$) and ts is the sampling time in $[s]$.

The following equation can be used to estimate FR_t, the horsepower reduction factor, based on current and proposed operating pressures (¨*Compressed air and Gas hand-*

book,1961¨)

$$FR_t = \frac{((p_{dp} + p)/p)^{(k-1)/(kN)} - 1}{((p_{dc} + p)/p)^{(k-1)/(kN)} - 1} \tag{23}$$

p_{dp} is the absolute discharge pressure at proposed operating pressure conditions (the relative discharge pressure is reported in `reduced_pressure`[4]), p_{dc} is the absolute discharge pressure at current pressure conditions (the relative one is reported in `p_out`), p is the inlet pressure (in `pressure_amb`), k the specific air heat and N the number of stages (stored in `stages_number`)

3.10.1 Input/Output Signals

Input signal:

- air pressure in $[Pa_g]$ evaluated at the before the pressure reducer.
- compressor power in load mode in $[kW]$.
- compressor flow in $[m^3/s]$. I is not visible because it isn't a real input signal but a value required to evaluate other signals.

Output signal: energy waste in kWh.

[4]the absolute value corresponds to the sum of the relative and ambient pressure

3.11 Flow profile

The flow profile module (Figure (11)) receives, as input, the pressure measured upstream this block: this signals is not used inside the block flow profile; it was created only to give a logical sense to the final compressed air system. The block has not output signals.

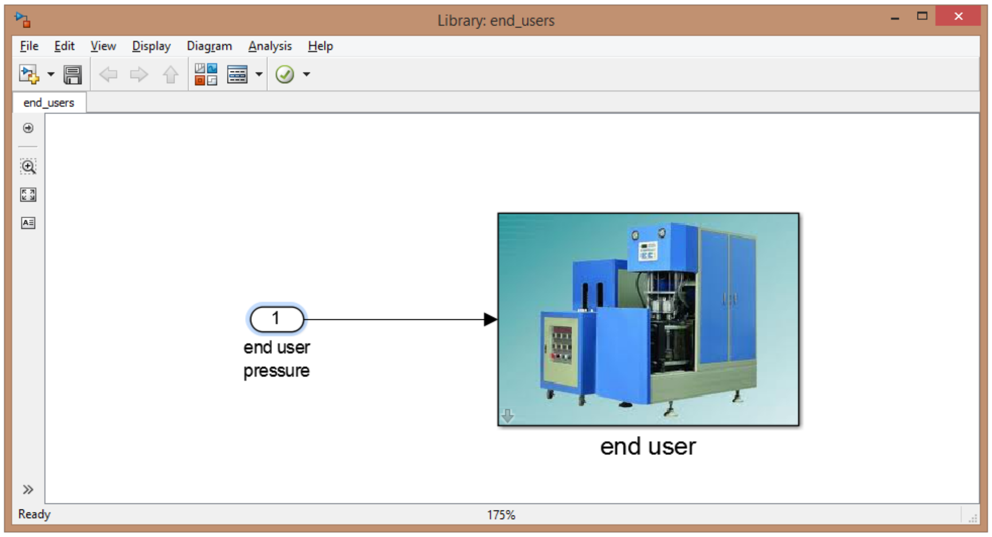

Figure 11: Module representing the flow required by the end-users.

This module was create only to contains the vector of flows required by the end-users: these values are contained inside the constant `flow_out`.

3.12 System

The block reported in Figure (12) not receive signals as input and not provide signal as output: the only functionality of this block is to help the users to set parameters describing the environment surrounding the compressed air system.

The values is possible to set are the following:

- `pressure_amb`: it contains the ambient pressure expressed in $[Pa]$ absolute. The default value is 101325 Pa.
- `T_amb`: it represents the temperature in $[K]$ of the inlet air of the compressor. The default value is 298.15 K
- `cp`: it is the specific heat at constant pressure expressed in $[J/(kg\ K)]$. The default value is 1005 $J/(kg\ K)$.
- `cv`: it is the specific heat at constant volume expressed in $[J/(kg\ K)]$. The default value is 718 $J/(kg\ K)$.
- `R_aria`; this parameter represents the air's constant; as default, this value assume the value of 287 $J/(kg\ K)$.

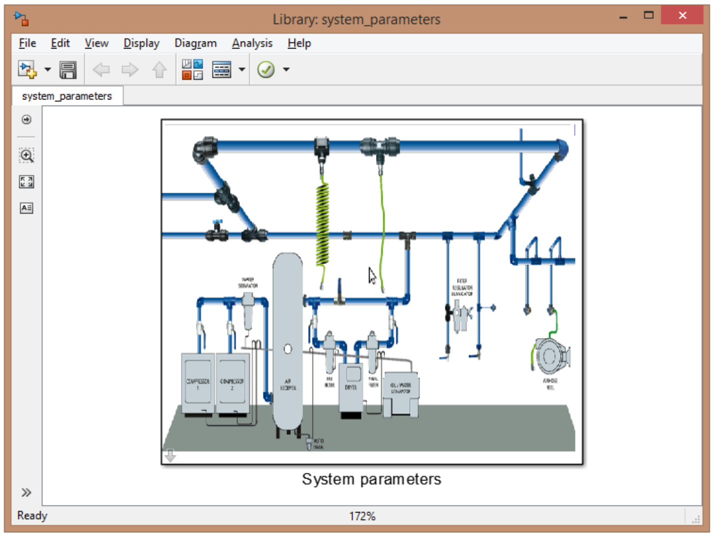
Library: system_parameters
File Edit View Display Diagram Analysis Help
system_parameters
System parameters
Ready
172%

Figure 12: System parameter module.

4 How to set up a simulation

To properly use the libraries, it is firstly necessary to set their paths as follows:

`File > Set Path > Add With Subfolders > .../simulator`

The compressed air system layout we want to simulate must be reported inside a Matlab `model`: to open this kind of file follow the instruction:

`File > New > Model`

Now it is possible to start to reproduce the compressed air system; inside the folder `devices` there are the modules described in sections before: to simulate one of that, you must open the related library, copy the block and paste inside the `model` opened before. Once reported all the blocks, connect properly their input/output signals.

Then it is necessary to set all that parameters will be used during the simulation: inside the file `parameters.m` all the simulation's parameters are reported; if it is necessary you can modify the default value; after that you must run the file `parameters.m`: the variables will appear in your Matlab WorkSpace. To modify these values, it is also possible to double click on the blocks: it will open a mask where it is possible to manually insert the values of the parameters required. If you do not modify them, they will assume the value reported inside the variable specified inside the mask.

Now, it is possible to run the simulation as follows:

`Simulation > Start`

In order to understand how ModSCA works, an easy example is proposed. The example `LABAC.slx` represents model describing the compressed air system installed in our laboratory (LABAC: `http://www-3.unipv.it/energy/labac`). It is composed by:

- `load-unload compressor`
- `receiver`
- `filter`
- `pipe network`
- `end user`

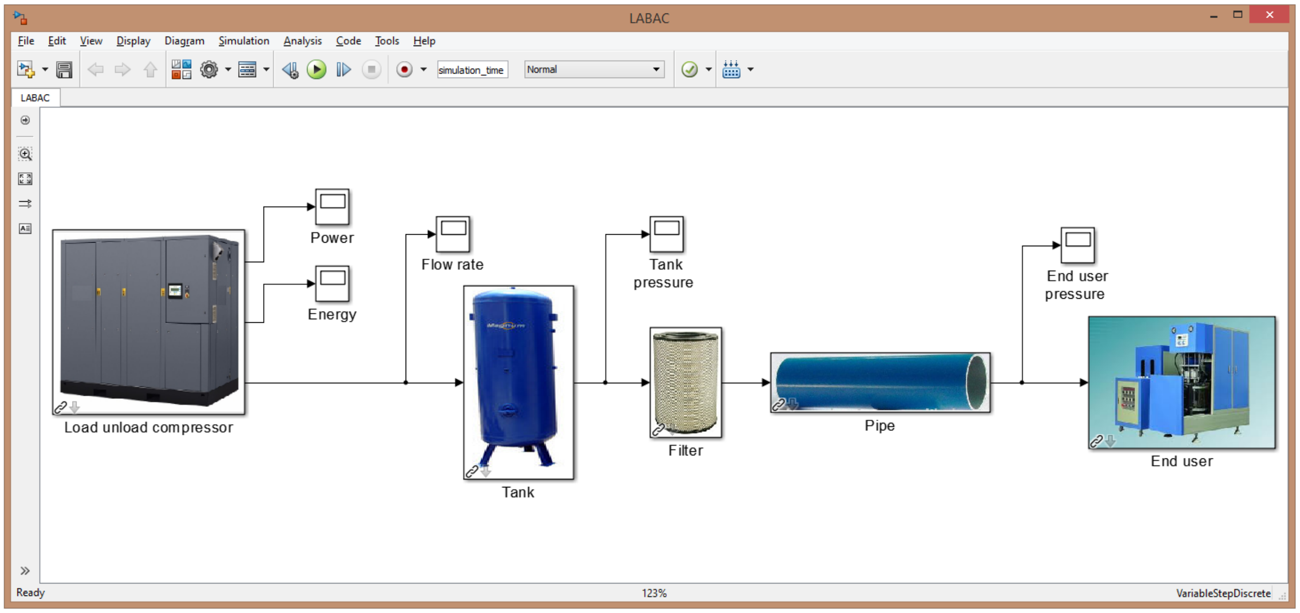

Figure 13: LABAC.

.

A Sub-blocks (minor importance)

In `simulator/subsystems/compressors`

- `energy.slx`

$$\sum_{t=1}^{Ts} P(t) * ts \tag{24}$$

  where Ts is the simulation time (in `simulation_time`), and $P(t)$ the instantaneous power absorbed by the compressor and ts the sampling period.

- `reduction_motor_speed.mdl`: it implements the motor speed variation when the compressor switch from load to off and vice versa. In the first case, the motor speed is reduced each ts of Δn_{max} until the speed reaches the value zero. In the other case, the speed increase with the same velocity from zero to n_{min}.

- `rendimento_politropica`: this sub-block calculates the polytropic efficiency as follows

$$\eta_{pol} = \frac{\frac{n}{n-1}}{\frac{k}{k-1}} \tag{25}$$

  where n is the polytropic coefficient and k is c_p/c_v.

In `simulator/subsystems/air_dryer`

- `saturated_vapour.slx`.
  This sub-block calculates the ideal amount of water vapour contained in the compressed air after the air dryer. The block interpolates the data of the water vapour contained in one cubic meter of air at 5 bar and at 8 bar for different temperatures.

In `simulator/subsystems/constants`

- `coefficiente_politropico.slx`

  Considering the air as an ideal gas, the thermodynamic process inside the compressor follows a polytropic equation:

$$\frac{T_2}{T_1} = (\frac{P_2}{P_1})^{\frac{n-1}{n}} \tag{26}$$

  where n is the polytropic coefficient, T_1 and T_2 represent the air temperature respectively before and after the compression process; these two values are reported in

`T_ref_in` and `T_ref_out`. P_1 and P_2 the air pressure before and after the compression process (`pres_ref_in` and `pres_ref_out`). From Equation (26) follows

$$n = \frac{1}{1 - \frac{log\left(\frac{T_2}{T_1}\right)}{log\left(\frac{P_2}{P_1}\right)}} \tag{27}$$

- `reynolds.slx`

  In fluid mechanics, the Reynolds number (Re) is a dimensionless number that gives a measure of the ratio of inertial forces to viscous forces and consequently quantifies the relative importance of these two types of forces for given flow conditions.

  $$Re = \frac{D\rho v}{\mu} \tag{28}$$

  where D is the pipe's diameter, ρ is the air density, v the air velocity and μ the absolute viscosity whose value is calculated in a dedicated sub block describe below.

In `simulator/subsystems/parameter`

- `air_density.slx`

  The air density can be calculated from the ideal gas law $pv = mRT$

  $$\rho = \frac{m}{v} = \frac{p}{RT} \tag{29}$$

  where p is the absolute air pressure, R the air constant ant T the air temperature.

- `fad_loss.slx` For each sample time, it contains the volume flow rate lost along the distribution system.

  $$M_l = C_d \cdot A \cdot \rho_i \sqrt{2 \cdot \frac{k-1}{k} \cdot \frac{p_i}{\rho_i} \cdot (\frac{p_a}{p_i})^{\frac{2}{k}} \cdot [1 - (\frac{p_a}{p_i})^{\frac{k-1}{k}}]} \tag{30}$$

  Where C_d is the emission coefficient, A is the area of an equivalent hole, ρ_i is the density of air inside the pipe while ρ_a is the one outside the pipe, p_i is the pressure inside the pipe while ρ_a is the one outside the pipe. After computing the mass flow lost $[kg/s]$, we need to pass in volume flow rate expressed in FAD condition $[m^3/s]$.

  $$q_l = \frac{M_l \cdot R \cdot T}{p} \tag{31}$$

  where $p = 100000\ Pa$, $T = 293.15K$ for the FAD condition.

- `fad_to_kgs.slx`

  This sub block calculates the mass flow provided by the compressor starting from the FAD (Free Air Delivery): it is the amount of atmospheric air (free air) can be sucked in by the compressor at inlet conditions: $p = 100000\ Pa$, $T = 20°C$ and $U\% = 0$.

$$\dot{m} = \frac{10^5 q_c}{R * 293.15K} \tag{32}$$

- `fad_to_m3s.slx`

  This sub block calculates the compressor's flow at operative condition starting from the FAD conditions: the first step to implement is to calculates the mass flow from the FAD as reported in Equation (32); then, it is possible to convert the mass flow to volumetric flow at operative conditions. The following equation allows to directly calculate the volumetric flow at operative conditions from FAD conditions.

$$\dot{v} = \frac{10^5 q_c T_{out}}{R p_{out} * 293.15K} \tag{33}$$

  T_{out} and p_{out} are respectively the temperature and pressure at operative conditions.

- `filter_p_drop.slx`

  This sub block calculates the pressure drop due to the filter dust cake at each sample time. The following equation allows to directly calculate the pressure drop:

$$\Delta p = \frac{150 \cdot q \cdot v \cdot \rho_a \cdot \mu}{A \cdot \rho_p \cdot d_p^2} \frac{(1-\epsilon)}{\epsilon^3} \tag{34}$$

  where q is the volume flow rate, v is the air speed, μ is the dynamic viscosity of the air, ρ_a is the air density while ρ_p is the density of the particles. A is the filter area and d_p is the diameter of the particles.

- `time_load_unload_stop.slx`: this sub block calculates each sample time the total load, unload and off time of the compressor from the beginning of the simulation.

- `viscosita_dinamica_aria.slx` This sub-block calculates the air absolute viscosity: it can can be defined as the resistance to flow encountered when one layer or plane of fluid attempts to move over another identical layer or plane of fluid at a given speed. Absolute viscosity is also called dynamic viscosity. The absolute viscosity can be obtained from dedicated graphs: only for compressed air used for industrial purpose can be used the following empirical equation:

$$\mu = 1.458\ 10^{-6} \frac{T^{1.5}}{T + 110.4} \tag{35}$$

The Equation (35), used in the Directive 2005/78/CE, varies according to the air temperature. The measurement unit is $Pa\ s$.

B file `parameters.m`

Table 3:

Constant's name	Unit	Default value	Description	Variable
`simulation_time`	$[s]$	604800	Total simulation time	T_s
`Ts`	$[s]$	1	Sample time	ts
`cp`	$[J/(kgK)]$	1005	Constant pressure specific heat	c_p
`cv`	$[J/(kgK)]$	718	Constant volume specific heat	c_v
`R_aria`	$[J/(kgK)]$	287	Air constant	R
`T_amb`	$[K]$	293.15	Air ambient temperature	T_{amb}
`T_air_out`	$[K]$	300.15	Temperature of compressed air leaving the compressor	
`pressure_amb`	$[Pa]$	101325	Ambient pressure	p_{in}
`U_rel`	%	60	Relative humidity	

Table 4:

Constant's name	Unit	Default value	Description	Variable
`fad`	$[m^3/s]$	0.043	Compressor flow provided by the compressor at fad conditions	
`t_on_off`	$[s]$	120	Time that must elapse before the compressor can starts again, from stop mode	
`n_avviamenti`	$[starts/h]$	120	Maximum number of starts per hour	
{`fraction_power_` `_unload`}	%	0.302	Percentage of rated power absorbed by the compressor in unload mode	
`rendimento_motore`	%	0.9	Compressor's motor efficiency	η_m
{`rendimento_` `_trasmissione`}	%	0.935	Compressor's gear efficiency	η_{gear}
{`rendimento_` `_politropica`}	%	0.66	Polytropic compression efficiency	η_{pol}
`time_from_off_to_on`	$[s]$	33	Time in unload mode before a compressor can start again	
`fan_power`	$[kW]$	0.7	Power absorbed by the fun installed inside the compressor	P_{fun}
`oil_pump_power`	$[kW]$	0	Power absorbed by the oil pump installed inside the compressor	$P_{oil\,pump}$
{`velocity_` `_loadTOunload`}	$[s]$	16.93	Time that must elapse before the power absorbed from the grid passes from load to unload power	
`T_ref_out`	$[K]$	358.15	Reference outlet air temperature (without oil)	
`T_ref_in`	$[K]$	293.15	Reference inlet air temperature	
`pres_ref_out`	$[Pa_g]$	950000	Reference outlet air pressure (relative)	
`pres_ref_in`	$[Pa]$	100000	Reference inlet air pressure (absolute)	
`pres_ref`	$[Pa_g]$	950000	Relative pressure whose temperature is `T_air_compressor_out`	
{`prefilter_RSC_` `_pressure_drop`}	$[Pa]$	0	Pressure drop caused by the load/unload compressor's filter	
{`prefilter_OS_` `_pressure_drop`}	$[Pa]$	0	Pressure drop caused by the oil separator filter	

Table 5:

Constant's name	Unit	Default value	Description	Variable
pipe_length	$[m]$	0	Length of the pipe	L
leaks	$[m^3/s]$	0	Amount of air flow leakage expressed in FAD	q_{leaks}
volume_tank	$[m^3]$	10	Volume of the receiver (the pipe volume is evaluated and then added to this value) V_r	

Table 6:

Constant's name	Unit	Default value	Description	Variable
pmin	$[Pa_g]$	600000	Load pressure	p_{load}
pmax	$[Pa_g]$	700000	Unload pressure	p_{unload}
p_out	$[Pa_g]$	700000	Compressed air pressure provided by the compressor	
p_iniziale_sistema	$[Pa_g]$	650000	Pressure inside the system at the beginning of the simulation	p_0

Table 7:

Constant's name	Unit	Default value	Description	Variable
{n_saracinesche_ _aperte}	-	0	Number of shutters opened	
{n_saracinesche_ _mezze_aperte}	-	0	Number of shutters half-opened	
n_valvole_membrana	-	0	Number of membrane valves	
n_curve	-	0	Number of curves	
n_T_dritta	-	0	Number of T-tube	
n_T_90	-	5	Number of angle pipe	
D_interno_tubo	$[m]$	0.05	Pipe's internal diameter	D
scabrezza_assoluta	$[m]$	0.00005	Pipe's absolute roughness	ϵ/D
T_air_enduser	$[K]$	293.15	Air temperature sampled near the end-users	

Table 8:

Constant's name	Unit	Default value	Description	Variable
`flow_out`	$[m^3/s]$	0.01	Flow rate required by the end-users at FAD conditions	q_{usr}

Table 9:

Constant's name	Unit	Default value	Description	Variable
`filter_pressure_ _drop`	$[Pa]$	20000	Filter's pressure drop	Δp_f
`T_air_after_dryer`	$[K]$	298.15	Compressed air temperature after the air passes through the air dryer	
`rendimento_air_ _dryer`	[%]	80	Air dryer efficiency	

Table 10:

Constant's name	Unit	Default value	Description	Variable
`kp`	-	20	PI proportional gain	k_p
`ki`	-	0.05	PI integral gain	k_i
`pressure_setpoint`	$[Pa_g]$	600000	Set-point of the ASD	p_{sp}
`p_load_vsd`	$[Pa_g]$	600000	ASD load pressure	
`p_off_vsd`	$[Pa_g]$	650000	ASD unload pressure	
`scaling_factor`	-	2.4e-04		
`max_delta_speed`	$[rad/s]$	30.4211	Maximum variation of the motor speed allowed during `Ts`	Δn_{max}
`max_speed`	$[rad/s]$	608.4218	Maximum motor speed for an ASD	n_{max}
`min_speed`	$[rad/s]$	225.116	Minimum motor speed for an ASD	n_{min}
`max_flow`	$[m^3/s]$	0.06	Maximum flow provided by an ASD	q_{max}
`min_flow`	$[m^3/s]$	0.0205	Minimum flow provided by an ASD	q_{min}
`q_sc`	-	0.0601	Coefficient	
`p_sc`	-	1350000	Coefficient	
`P_sc`	-	2.4712e+04	Coefficient	
`a_P0`	-	0.0411	Coefficient describing relation between the flow provided by an ASD and the absorbed power	
`a_P1`	-	0.3124	Coefficient describing relation between the flow provided by an ASD and the absorbed power	
`a_P2`	-	0.0395	Coefficient describing relation between the flow provided by an ASD and the absorbed power	
`a_P3`	-	1.1822	Coefficient describing relation between the flow provided by an ASD and the absorbed power	
`a_P4`	-	0.2422	Coefficient describing relation between the flow provided by an ASD and the absorbed power	
`a_P5`	-	-0.5426	Coefficient describing relation between the flow provided by an ASD and the absorbed power	

Table 11:

Constant's name	Unit	Default value	Description	Variable
`stages_number`	-	1		
`reduced_pressure`	$[Pa_g]$	50000		