HVDC Stability Simulator

Developing Guide

Aleksandra Lekić
ELECTA and EnergyVille,
KU Leuven, Leuven, Belgium
aleksandra.lekic@kuleuven.be

November 19, 2019

Contents

1 Multiport ABCD representation

The system is presented as interconnection of the components. For the simplicity of calculations of the transfer functions and/or input and output impedance, each component is modeled as a multiport network as depicted in the Fig. 1. Input voltages and currents are vectors denoted as $\mathbf{V}_p$ and $\mathbf{I}_p$, while output voltages and currents are $\mathbf{V}_s$ and $\mathbf{I}_s$. Generally, multiport network has the same number of input and output ports and thus, the dimensions of the vectors are the same, denoted as n.

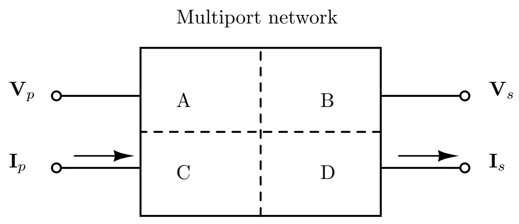


Figure 1: Multiport network.

Multiport network can be represented with ABCD parameters, where each of parameters $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ represent $n \times n$ matrices and

$$\begin{bmatrix} \mathbf{V}_p \\ \mathbf{I}_p \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \times \begin{bmatrix} \mathbf{V}_s \\ \mathbf{I}_s \end{bmatrix} \tag{1}$$

.

As in the power system, components are interconnected, two possibilities for their connection are series and parallel connection.

- series connection of two multiport networks is depicted in Fig. 2. ABCD multiport representation is specially desirable for this type of connection, because the new parameters are determined in a simple matter as follows.

$$\begin{bmatrix} \mathbf{V}_p \\ \mathbf{I}_p \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ \mathbf{C}_1 & \mathbf{D}_1 \end{bmatrix} \times \begin{bmatrix} \mathbf{A}_2 & \mathbf{B}_2 \\ \mathbf{C}_2 & \mathbf{D}_2 \end{bmatrix}}_{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}} \times \begin{bmatrix} \mathbf{V}_s \\ \mathbf{I}_s \end{bmatrix} \tag{2}$$

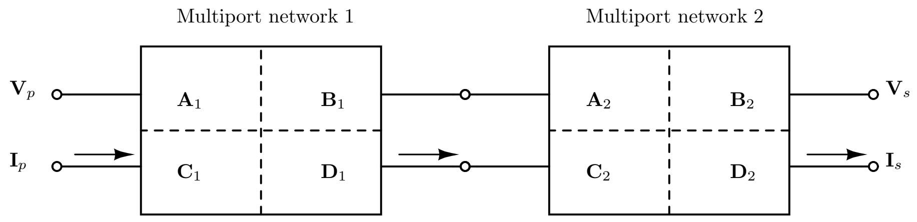


Figure 2: Series connected multiport networks.

- Parallel connection, depicted in the Fig. 3, is more complex for calculation. In the case of nonzero matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ (it is the same for the network whose matrices are with dimension 1×1, or two port networks) the parallel connection is represented as:

$$\begin{bmatrix} \mathbf{V}_p \\ \mathbf{I}_p \end{bmatrix} = \underbrace{\begin{bmatrix} (\mathbf{B}_1^{-1} + \mathbf{B}_2^{-1})^{-1} (\mathbf{B}_1^{-1}\mathbf{A}_1 + \mathbf{B}_2^{-1}\mathbf{A}_2) & (\mathbf{B}_1^{-1} + \mathbf{B}_2^{-1})^{-1} \\ \mathbf{C}_1 + \mathbf{C}_2 + (\mathbf{D}_2 - \mathbf{D}_1)(\mathbf{B}_1 + \mathbf{B}_2)^{-1}(\mathbf{A}_1 - \mathbf{A}_2) & \mathbf{D}_1 + (\mathbf{D}_2 - \mathbf{D}_1)(\mathbf{B}_1 + \mathbf{B}_2)^{-1}\mathbf{B}_1 \end{bmatrix}}_{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}} \times \begin{bmatrix} \mathbf{V}_s \\ \mathbf{I}_s \end{bmatrix}. \tag{3}$$

In case that some of the matrices is not invertible, then the previous equation becomes:

$$\begin{bmatrix}\mathbf{V}_p\\ \mathbf{I}_p\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{A}_i & \mathbf{0}\\ \mathbf{C}_1+\mathbf{C}_2+(\mathbf{D}_2-\mathbf{D}_1)\,\mathbf{B}_j^{-1}(\mathbf{A}_1-\mathbf{A}_2) & \mathbf{D}_i\end{bmatrix}}_{\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{bmatrix}} \times \begin{bmatrix}\mathbf{V}_s\\ \mathbf{I}_s\end{bmatrix}, \tag{4}$$

where $i, j \in \{1, 2\}$ and i denotes invertible matrix $\mathbf{B}_i$ and $j \neq i$.

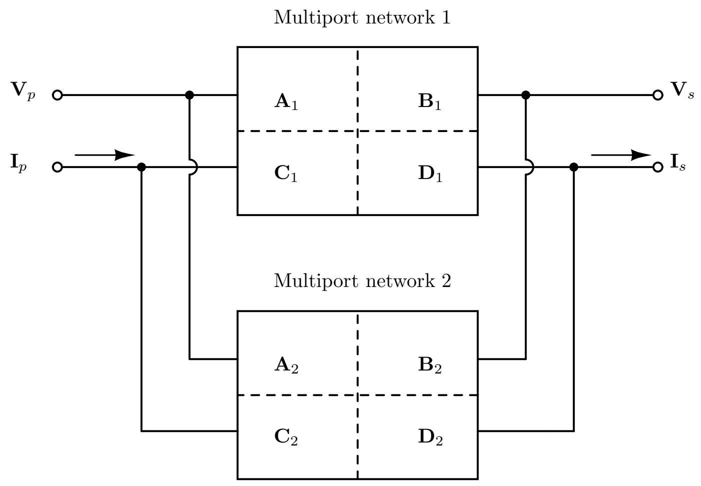


Figure 3: Parallel connected multiport networks.

1.1 Determining input/output impedance of the network

Let us assume that every output port, represented with the voltage V_{si} and the current I_{si}, is closed with an impedance Z_{ti}. Then we can write the equation $V_{si} = Z_{ti}\, I_{si}$, or in the matrix form

$$\mathbf{V}_s = \mathbf{Z}_t \odot \mathbf{I}_s = \widetilde{\mathbf{Z}}_t \times \mathbf{I}_s, \tag{5}$$

for $\odot$ denoting hadamard product, and $\mathbf{Z}_t$ being corresponding closing impedance column vector, and $\widetilde{\mathbf{Z}}_t = \mathrm{diag}\{\mathbf{Z}_t\}$.

Input impedance can be then estimated from equation

$$\begin{bmatrix}\mathbf{V}_p\\ \mathbf{I}_p\end{bmatrix} = \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{bmatrix} \times \begin{bmatrix}\widetilde{\mathbf{Z}}_t \times \mathbf{I}_s\\ \mathbf{I}_s\end{bmatrix},$$

as

$$\mathbf{Z}_p = (\mathbf{A} \times \widetilde{\mathbf{Z}}_t + \mathbf{B}) \times (\mathbf{C} \times \widetilde{\mathbf{Z}}_t + \mathbf{D})^{-1}. \tag{6}$$

Similarly, by closing input ports with a diagonal impedance $\widetilde{\mathbf{Z}}_t$, from the output ports can be estimated impedance

$$\mathbf{Z}_s = (\widetilde{\mathbf{Z}}_t \times \mathbf{C} - \mathbf{A})^{-1} \times (\widetilde{\mathbf{Z}}_t \times \mathbf{D} - \mathbf{B}). \tag{7}$$

1.2 Idea for determining the port impedance

The ABCD parameters can be used to determine the impedance “visible” from the desired node or a component. To determine the impedance “visible” from the desired node/nodes in the system, recursively is formed the smaller partition of the system, containing only the nodes and the components included in the path between desired nodes.

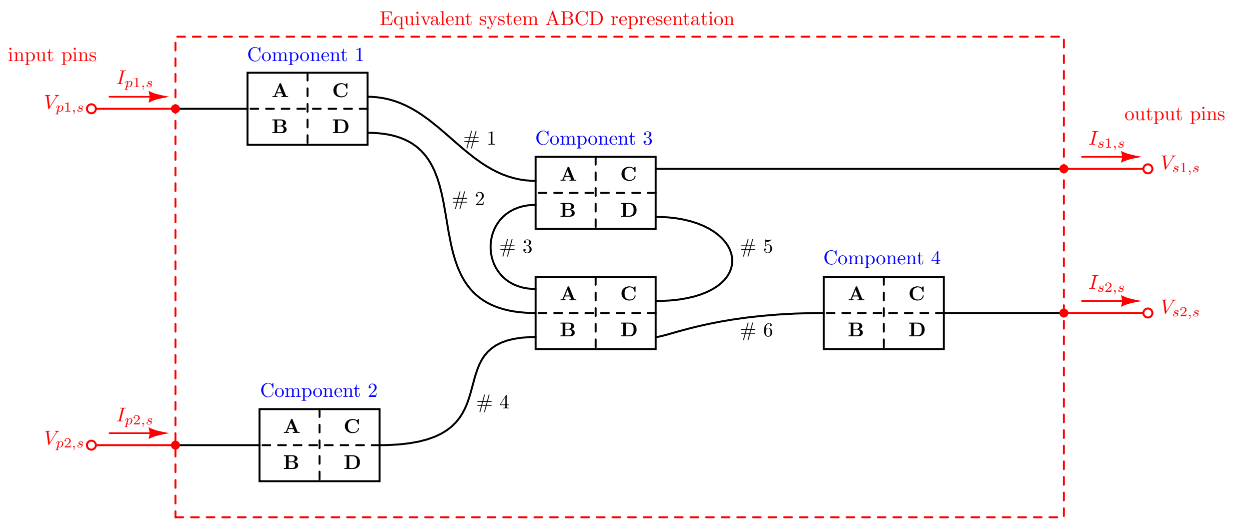


Figure 4: Model of the polyphase subsystem.

The example of the obtained subsystem is depicted in the Fig. 4, where the nodes denoted as $V_{p1,s}$, $V_{p2,s}$ and $V_{s1,s}$, $V_{s2,s}$ represent 2×2 port whose equivalent impedance should be determined as a 2×2 impedance matrix.

As can be seen from the Fig. 4, the obtained polyphase subsystem contains m components, where each component i is represented with with p_p^i inputs and p_s^i outputs. Subsystem also contains n_n nodes (both input and output), n_o nodes being output nodes (denoted as $V_{s1,s}$ and $V_{s2,s}$ in Fig. 4) or ground nodes, and n_c input currents (denoted as $I_{p1,s}$ and $I_{p2,s}$ in Fig. 4).

Let us assume the following naming convention. The ith component has input voltages and currents denoted as $\mathbf{V}_{pi}$ and $\mathbf{I}_{pi}$ (current enters the component and exits the node), and output voltages and currents $\mathbf{V}_{si}$ and $\mathbf{I}_{si}$ (current exits the component and enters the node). $\mathbf{I}_i$ are the input subsystem currents entering the subsystem and exiting the input nodes, and $\mathbf{I}_0$ are the currents through ground(s) and the output pins, which enter the node.

Then, the set of $n_n + \sum_{i=1}^{m} 2p_p^i$ equations can be written:

- n_n equations for each of the nodes inside the network denoting the sum of the currents exiting the node. The positive direction for the node cutset is chosen as the current exiting the node.
- $\sum_{i=1}^{m} p_p^i$ equations giving ABCD component relationship between input voltages $\mathbf{V}_{pi}$ and output voltages $\mathbf{V}_{si}$, and currents $\mathbf{I}_{si}$.
- $\sum_{i=1}^{m} p_p^i$ equations giving ABCD component relationship between input currents $\mathbf{I}_{pi}$ and output voltages $\mathbf{V}_{si}$, and currents $\mathbf{I}_{si}$.

The unknown variables are $n_v = n_n - n_o$ node voltages, n_c input currents denoted as $\mathbf{I}_i$ and $\sum_{i=1}^{m} p_p^i + p_s^i$ component currents $\mathbf{I}_{pi}$ and I_{si}.

The complete set of equations is written in the matrix form and it consists of $n_n + \sum_{i=1}^{m} 2p_p^i$ equations with $n_v + n_c + \sum_{i=1}^{m}(p_p^i + p_s^i)$ variables and matrix of outputs $\left(n_n + \sum_{i=1}^{m} 2p_p^i\right) \times 2n_o$. It is:

$$\mathbf{M} \times \mathbf{X} = \mathbf{N} \times \mathbf{Y} \tag{8}$$

where the matrices $\mathbf{M}$ and $\mathbf{N}$ consist of numerical and symbolic coefficients, vector $\mathbf{X} = \begin{bmatrix} V_1 & \cdots & V_{n_v} & I_{i1} & \cdots & I_{in_c} & \mathbf{I}_{p1} & \mathbf{I}_{s1} & \cdots & \mathbf{I}_{pm} & \mathbf{I}_{sm} \end{bmatrix}$ consists of the unknown variables and vector $\mathbf{Y} = \left[\mathbf{V}_{0j}, \mathbf{I}_{0j}\right]^T \Big|_{j=1}^{n_o}$ of the output and ground voltages and currents. The solution of the previous system (8) is given as reduced row echelon form of concatenated matrices $[\mathbf{M}, \mathbf{N}]$.

2 Implementation of the components

The main components of the HVDC power system are DC and AC grid (or sources), impedances, transformers, transmission lines and cables, breakers and MMC converters. Each of the components will be presented as multiport network using ABCD parameters as follows.

2.1 Impedance

Impedance can be formed between n input ports and n output ports. Example of the impedance with two input ports and two output ports is given in the Fig. 5. Impedance is defined with $n \times n$ matrix $\mathbf{Z}$, where

$$\mathbf{Z} = \begin{bmatrix} Z_{11} & Z_{12} & \cdots & Z_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{n1} & Z_{n2} & \cdots & Z_{nn} \end{bmatrix}$$

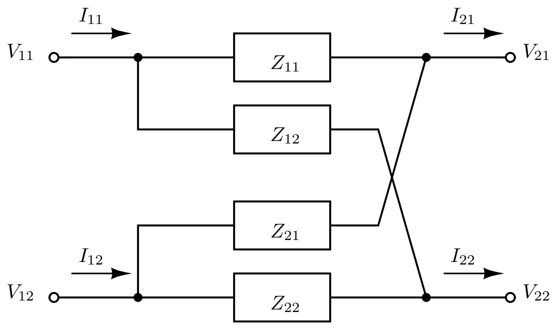


Figure 5: Model of the 2 input ports/2 output ports impedance.

To represent impedance as a multiport component with ABCD parameters, it is required to solve the following equation constructed using MNA (Modified Nodal Analysis) approach.

$$\underbrace{\left[\begin{array}{c|c} \underset{i}{\operatorname{diag}}\left\{\sum\limits_{j, Z_{ij} \neq 0} \frac{1}{Z_{ij}}\right\}_{n\times n} & \operatorname{diag}\{-1\}_{n\times n} \\ \hline \mathbf{N}_{1,n\times n} & \mathbf{0}_{n\times n} \end{array}\right]}_{\mathbf{M}_1} \times \begin{bmatrix} \mathbf{V}_p \\ \mathbf{I}_p \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} \mathbf{N}_{2,n\times n} & \mathbf{0}_{n\times n} \\ \hline -\underset{i}{\operatorname{diag}}\left\{\sum\limits_{j, Z_{ji} \neq 0} \frac{1}{Z_{ji}}\right\}_{n\times n} & \operatorname{diag}\{-1\}_{n\times n} \end{array}\right]}_{\mathbf{M}_2} \times \begin{bmatrix} \mathbf{V}_s \\ \mathbf{I}_s \end{bmatrix}, \tag{9}$$

where matrices $\mathbf{N}_1$ and $\mathbf{N}_2$ consist of n rows with n columns with entries at the position (i, j) equal to $-\frac{1}{Z_{ji}}$ and $\frac{1}{Z_{ij}}$, for $Z_{ij,ji} \neq 0$ (where i represents row and j column in impedance matrix), respectively. The solution of the previous system is given as $\mathbf{M}_1^{-1}\mathbf{M}_2$ if $\mathbf{M}_1$ is invertible matrix, or by determining LU decomposition and reduced row echelon form if it is not invertible.

For example, for the circuit depicted in Fig. 5, the equations would be:

$$\left[\begin{array}{cc|cc} \frac{1}{Z_{11}} + \frac{1}{Z_{12}} & 0 & -1 & 0 \\ 0 & \frac{1}{Z_{21}} + \frac{1}{Z_{22}} & 0 & -1 \\ \hline -\frac{1}{Z_{11}} & -\frac{1}{Z_{21}} & 0 & 0 \\ -\frac{1}{Z_{12}} & -\frac{1}{Z_{22}} & 0 & 0 \end{array}\right] \times \begin{bmatrix} V_{11} \\ V_{12} \\ I_{11} \\ I_{12} \end{bmatrix} = \left[\begin{array}{cc|cc} \frac{1}{Z_{11}} & \frac{1}{Z_{12}} & 0 & 0 \\ \frac{1}{Z_{21}} & \frac{1}{Z_{22}} & 0 & 0 \\ \hline -\left(\frac{1}{Z_{11}} + \frac{1}{Z_{21}}\right) & 0 & -1 & 0 \\ 0 & -\left(\frac{1}{Z_{12}} + \frac{1}{Z_{22}}\right) & 0 & -1 \end{array}\right] \times \begin{bmatrix} V_{21} \\ V_{22} \\ I_{21} \\ I_{22} \end{bmatrix}.$$

2.2 Transformer

Transformer is modeled as in the Fig. 6. Parameters of the transformer can be defined explicitly or determined from the on-site test data as described in [1]. On-site test data is given in the form of open and short-circuit values of the primary side voltage V_1 and current I_1 and secondary side voltage V_2 and current I_2, then the core $P_{1,core}$ and winding $P_{1,winding}$ power loss.

Then the parameters from Fig. 6 can be estimated as:

$$\begin{aligned}
R_{ps} &= \frac{P_1^{short}}{(I_1^{short})^2}, \\
L_{ps} &= \frac{Q_1^{short}}{\omega(I_1^{short})^2}, \\
R_m &= \frac{(V_1^{open})^2}{P_1^{open}}, \\
L_m &= \frac{(V_1^{open})^2}{\omega Q_1^{open}}, \\
n &= \frac{V_1^{open}}{V_2^{open}}, \\
R_p &= \frac{R_{ps}}{2}, \\
R_s &= \frac{R_{ps}}{2n^2}, \\
L_p &= \frac{L_{ps}}{2}, \\
L_s &= \frac{L_{ps}}{2n^2}, \qquad (10)
\end{aligned}$$

knowing that $Q_1^{o,s} = \sqrt{(V_1^{o,s} I_1^{o,s})^2 - P_1^{o,s2}}$.

ABCD multiport parameters are then estimated as:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \mathbf{Y}_{turn} \times \left(\mathbf{Z}^p_{winding} \times \mathbf{Y}_{iron} \times \mathbf{N}_{tr} \times \mathbf{Z}^s_{winding} \parallel \mathbf{Z}_{stray}\right) \times \mathbf{Y}_{turn}, \qquad (11)$$

where $\mathbf{Y}_{turn} = \begin{bmatrix} 1 & 0 \\ sC_t & 1 \end{bmatrix}$, $\mathbf{Z}^p_{winding} = \begin{bmatrix} 1 & sL_p + R_p \\ 0 & 1 \end{bmatrix}$, $\mathbf{Y}_{iron} = \begin{bmatrix} 1 & 0 \\ \frac{1}{sL_m} + \frac{1}{R_m} & 1 \end{bmatrix}$, $\mathbf{Z}^s_{winding} = \begin{bmatrix} 1 & sL_s + R_s \\ 0 & 1 \end{bmatrix}$, $Z_{stray} = \begin{bmatrix} 1 & \frac{1}{sC_{stray}} \\ 0 & 1 \end{bmatrix}$ and $\mathbf{N}_{tr} = \begin{bmatrix} n & 0 \\ 0 & \frac{1}{n} \end{bmatrix}$, for n being turn ratio.

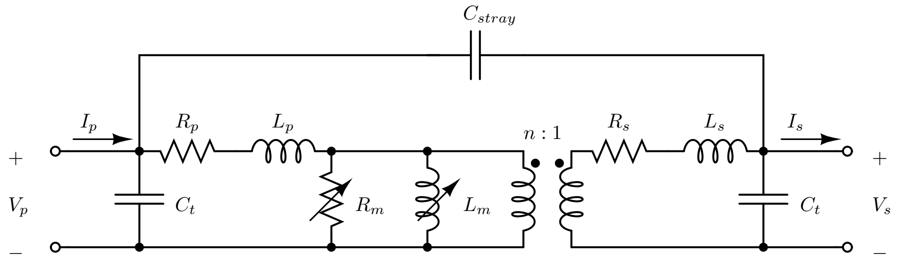


Figure 6: Model of the transformer.

At the output can be added serial load modeled as $Z_{load} = \begin{bmatrix} 1 & sL_{load} \\ 0 & 1 \end{bmatrix}$.

Three phase transformer can be in the YY and ΔY configuration, where each of the three transformers is represented by its equivalent from the Fig. 6.

- YY configuration is derived from the equation (11), such that

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \text{diag}\{A\}_{3\times3} & \text{diag}\{B\}_{3\times3} \\ \text{diag}\{C\}_{3\times3} & \text{diag}\{D\}_{3\times3} \end{bmatrix}.$$

- ΔY configuration is more complex and it is modeled using following equations. Inner primary and secondary stage of the transformer is given by

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}_{inner} = \mathbf{Z}^{p}_{winding} \times \mathbf{Y}_{iron} \times \mathbf{N}_{tr} = \begin{bmatrix} n + n\left(sL_p + R_p\right)\left(\frac{1}{sL_m} + \frac{1}{R_m}\right) & \frac{sL_p+R_p}{n} \\ n\left(\frac{1}{L_m} + \frac{1}{R_m}\right) & \frac{1}{n} \end{bmatrix}. \quad (12)$$

  Transformation Δ to Y transforms voltages from delta side $v_p^{a,b,c}$ to wye side voltages $v_s^{a,b,c}$ as $v_p^{a,b,c} = \sqrt{3}\, v_s^{a,b,c}$, while the currents are related as:

$$\mathbf{i}_p^{a,b,c} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 \\ 0 & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{bmatrix} \times \mathbf{i}_s^{a,b,c}.$$

  Using previous voltage/current relations and ABCD representation of the inner transfer function from expression (12) can be estimated as

$$\mathbf{Z}_{inner} = \left[\begin{array}{c|c} \mathrm{diag}\{A_{inner}\sqrt{3}\}_{3\times3} & \mathrm{diag}\{\frac{B_{inner}}{\sqrt{3}}\}_{3\times3} \\ \hline \mathrm{diag}\{C_{inner}\sqrt{3}\}_{3\times3} & \begin{matrix} \frac{D_{inner}}{\sqrt{3}} & 0 & -\frac{D_{inner}}{\sqrt{3}} \\ -\frac{D_{inner}}{\sqrt{3}} & \frac{D_{inner}}{\sqrt{3}} & 0 \\ 0 & -\frac{D_{inner}}{\sqrt{3}} & \frac{D_{inner}}{\sqrt{3}} \end{matrix} \end{array}\right]. \quad (13)$$

  Transformer is then represented using ABCD parameters as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \left(\mathbf{Y}_{turn} \times \left(\mathbf{Z}_{inner} \parallel \mathbf{Z}_{stray}\right) \times \mathbf{Y}_{turn}\right) \times \mathbf{Z}_{load}. \quad (14)$$

2.3 Transmission line

ABCD model parameters can be than estimated as in [2,3]

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \cosh(\Gamma l) & \mathbf{Y}_c^{-1}\sinh(\Gamma l) \\ \mathbf{Y}_c \sinh(\Gamma l) & \cosh(\Gamma l) \end{bmatrix} \quad (15)$$

where $\Gamma = \sqrt{\mathbf{Z}\mathbf{Y}}$ and $\mathbf{Y}_c = \mathbf{Z}^{-1}\,\Gamma$, and l being line or cable length.

2.3.1 Overhead line

Focusing on the PSCAD realization of the transmission line, see Fig. 7, the five possible realizations are defined: flat (horizontal as a flat without ground wires), vertical, delta (for lines with at least three phases), offset (at least three phases), concentric (at least three phases). Besides the conductor positions can be added manually as an absolute (x, y) positions.

Thus, the simulator enables creation of the overhead line with the **conductors** having the following fields:

- n_b - number of bundles (or phases)
- n_{sb} - number of subconductors per bundle
- y_{bc} - height of the lowest bundle above ground
- Δy_{bc} - vertical offset between bundles

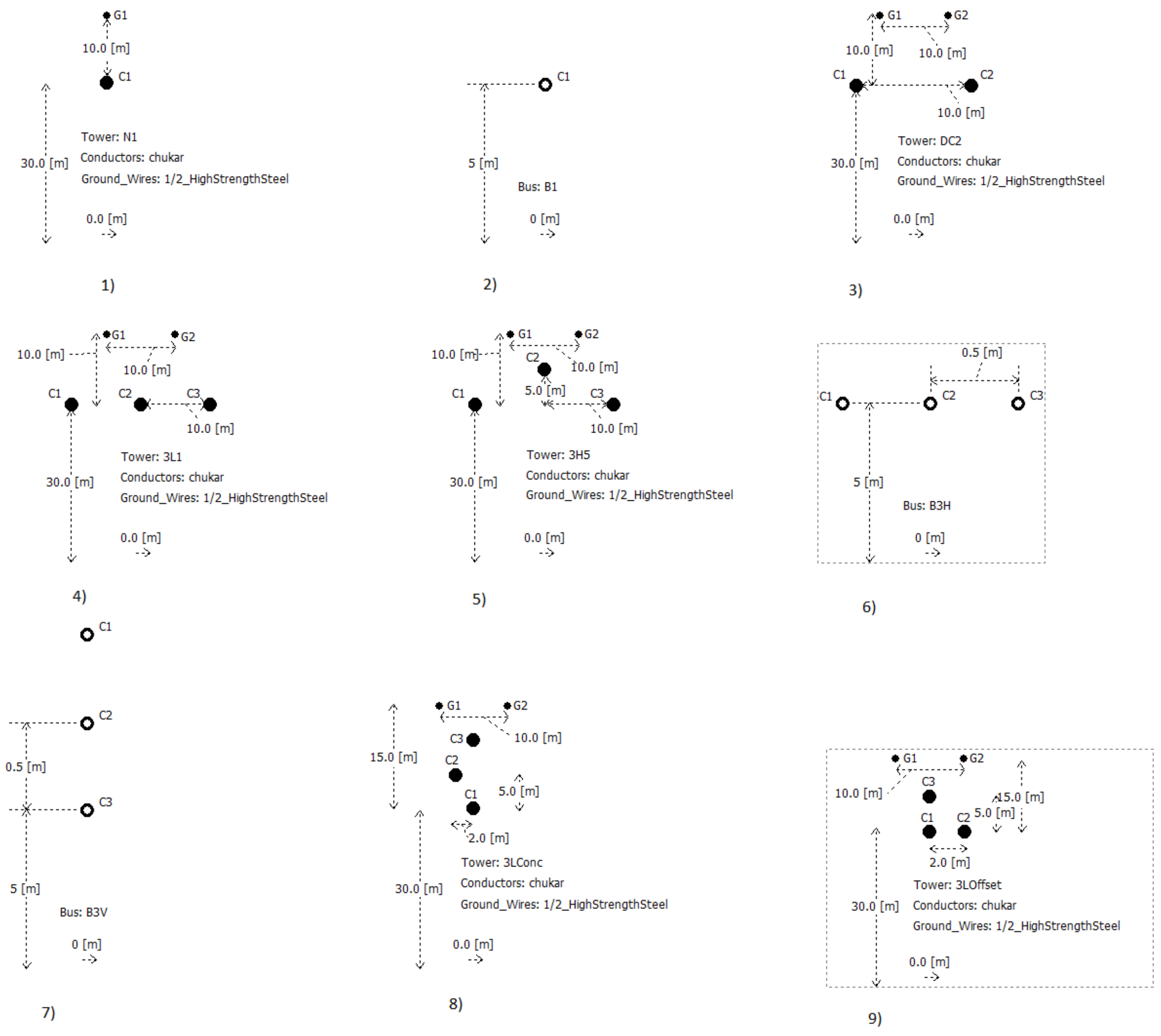


Figure 7: PSCAD overhead line orzanization: 1) single conductor with groundwire; 2) single conductor; 3) 2 conductors flat; 4) 3 conductors flat; 5) 3 conductors delta; 6) 3 conductors horizontal; 7) 3 conductors vertical; 8) 3 conductors concentric; 9) 3 conductors offset.

- Δx_{bc} - horizontal offset between the lowest bundles
- $\Delta \tilde{x}_{bc}$ - horizontal offset in the case of concentric and offset organization
- d_{sb} - distance between closest subconductors with equidistant concentric organization (symmetric)
- d_{sag} - maximal sag offset
- r_c - radius of the conductor
- R_{dc} - DC resistance of the conductor
- g_c - shunt conductance of the conductor
- $\mu_{r,c}$ - relative permeability of the conductor
- positions - added manualy

- organization which can be flat, vertical, concentric, delta and offset

Ground wires have the following properties:

- n_g - number of ground wires
- Δx_g - relative horizontal distance between ground wires
- Δy_g - relative vertical between ground wires and the lowest conductors
- r_g - radius of the ground wire
- $d_{g,sag}$ - ground wire sag
- $R_{g,dc}$ - DC resistance of the ground wires
- $\mu_{r,g}$ - relative permeability of the ground wire

Transmission line model is constructed using the procedure from [1,4]. The overhead transmission line consists of n_b conductors and n_g ground wires.

Each line/conductor positioned as x_c relatively starting from the central tower position and y_c vertically, measured from ground, with the sag at the midpoint between towers d_{sag}, see Fig. 8a. Thus, the modified vertical position is used in calculations as $\hat{y}_c = y_c - \frac{2}{3}\, d_{sag}$. Conductor is formed using n_{sb} sub-conductors grouped in the bundle, where all sub-conductors are grouped using symmetrical equidistant pattern with the distance between the two nearest sub-conductors being d_{bc}, or a bundle spacing. Using conductor position, the position of the each subconductor can be estimated. Knowing the angle between two sub-conductors on the circle and its radius

$$\begin{aligned}\varphi &= \frac{360°}{n_{sb}},\\ r &= \frac{d_{sb}}{2\sin(\varphi/2)},\end{aligned} \tag{16}$$

the position can be estimated starting from the angle $\varphi_s = \frac{\pi}{2}$ if number of sub-conductors is odd, or from $\varphi_s = \frac{\pi+\varphi}{2}$ for the even number of sub-conductors, as follows:

$$\begin{aligned}x_{bc} &= x_c + r\cos(\varphi_s + k\,\varphi),\\ y_{bc} &= y_c + r\sin(\varphi_s + k\,\varphi) - \frac{2}{3}\, d_{sag},\end{aligned} \tag{17}$$

for $k \in \{1, 2, ..., n_{bc}\}$. If number of sub-conductors is one than its position is $(x_c, \hat{y}_c)$. Each conductor is characterized with the relative permeability of the material μ_r, conductor dc resistance R_{dc} and the radius r_i.

Ground wires are modeled similarly, represented with their relative position (x_g, y_g), radius r_g, dc resistance R_{gdc} and the relative permeability of the material μ_r.

Earth parameters are given with permeability μ_e, permittivity ϵ_e and conductivity.

In order to represent transmission line using ABCD parameters, it is necessary to calculate series impedance and shunt admittance matrices [1]. Both matrices are of the size $n \times n$, where $n = \sum\limits_{i=1}^{n_c} n_{bc}^i + n_g$. Impedance matrix has the following form:

$$\mathbf{Z} = \mathrm{diag}(Z_i) + \begin{bmatrix} Z_{0,11} & \cdots & Z_{0,1n} \\ \vdots & \ddots & \vdots \\ Z_{0,n1} & \cdots & Z_{0,nn} \end{bmatrix} \tag{18}$$

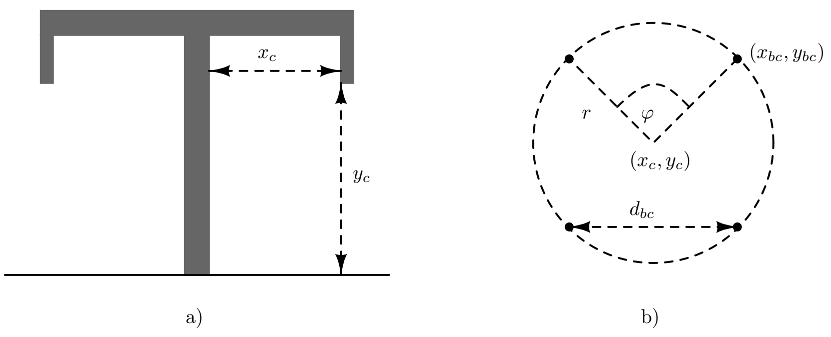


Figure 8: Overhead line modelling: a) tower and relative conductor positions; b) sub-conductor bundle.

where $Z_i = \frac{m\rho_i}{2\pi r_i}\coth(0.733mr_i) + \frac{0.3179\rho_i}{\pi r_i^2}$ for ith conductor/sub-conductor/ground wire and r_i is its radius, resistivity $\rho_i = R^i_{dc}\pi r_i^2$ and $m = \sqrt{j\omega\frac{\mu_0\mu_{r,i}}{\rho_i}}$; components $Z_{0,ij} = \frac{j\omega\mu_0}{2\pi}\log\left(\frac{\hat{D}_{ij}}{d_{ij}}\right)$ for

$$\begin{aligned}
d_{ij} &= \begin{cases} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, & i \neq j, \\ r_i, & i = j, \end{cases} \\
D_{ij} &= \begin{cases} \sqrt{(x_i - x_j)^2 + (y_i + y_j)^2}, & i \neq j, \\ 2y_i, & i = j, \end{cases} \\
\hat{D}_{ij} &= \sqrt{(y_i + y_j + 2d_e)^2 + (x_i - x_j)^2}, \\
d_e &= \sqrt{\frac{1}{j\omega\mu_e(\sigma_e + j\omega\epsilon_e)}}.
\end{aligned} \tag{19}$$

Shunt admittance is a matrix formed as

$$\mathbf{Y} = s\,\mathbf{P}^{-1} + \mathbf{G} \tag{20}$$

from matrix $\mathbf{P}$ with the components $\mathbf{P}_{ij} = \frac{1}{2\pi\epsilon_0}\log\left(\frac{D_{ij}}{d_{ij}}\right)$ and $\mathbf{G} = \mathrm{diag}\{g_c\}$.

2.3.2 Cable

Focusing on the available configurations of the cables in PSCAD, which enables cable organization inside the pipe, so called pipe-type cables, and cables placed underground. Cables are usually coaxial with up to 4 layers of both conductors and insulators.

Cables can be insulated or pipe-type coaxial cables. At the moment it is only implemented a group of coaxial cables. Group consists of n cables, each one have maximum three conducting layers and three insulation layers, as can be seen from Fig. 9. Conducting layers of the cable denoted as core, sheath and an armor. Between conducting layers are insulators, except for the last conductor where the insulator is not a necessity, but it is common. For each conductor are given parameters: r^c_i and r^c_o as conductor inner and outer radius in meters, conductor relative permeability μ^c_r and conductor resistivity ρ_c ([Ωm] is its unit). Insulator is described using parameters: r^i_i and r^i_o as insulator inner and outer radius in meters, ϵ^i insulator relative permittivity and μ^i_r insulator relative permeability.

Additional, the configuration parameters can be modified by adding two semiconducting layers in the insulator 1, and implementing the sheath consisting of the wire screen and outer sheath layer. Then the procedure described in [5] is applied.

- Conductor surface impedance

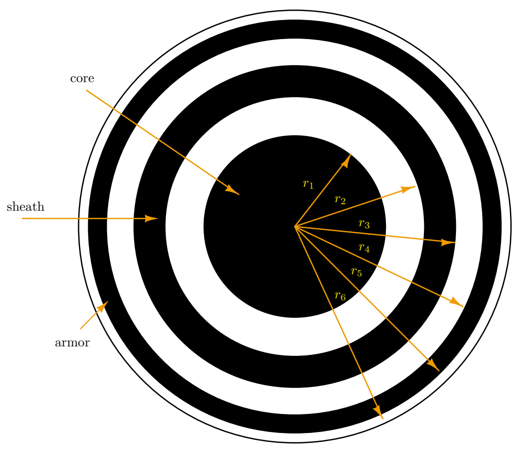


Figure 9: Coaxial cable.

Hollow conductor surface impedance is given by:

$$\begin{aligned} Z_{aa} &= \frac{\rho_c m}{2\pi r_i^c}\coth(m(r_o^c - r_i^c)) + \frac{\rho^c}{2\pi r_i^c\,(r_i^c + r_o^c)}\left[\frac{\Omega}{\mathrm{m}}\right] \quad \text{for inner surface,} \\ Z_{bb} &= \frac{\rho^c m}{2\pi r_o^c}\coth(m(r_o^c - r_i^c)) + \frac{\rho^c}{2\pi r_o^c\,(r_i^c + r_o^c)}\left[\frac{\Omega}{\mathrm{m}}\right] \quad \text{for outer surface,} \\ Z_{ab} &= \frac{\rho^c m}{\pi(r_i^c + r_o^c)}\operatorname{csch}(m(r_o^c - r_i^c))\left[\frac{\Omega}{\mathrm{m}}\right], \end{aligned} \tag{21}$$

where $m = \sqrt{j\omega\mu_r^c}$ For non-hollow conductor, the outer surface impedance is

$$Z_{bb} = \frac{\rho^c m}{2\pi r_o^c}\coth(0.733 m r_o^c) + \frac{0.3179\rho^c}{\pi r_o^{c2}}\left[\frac{\Omega}{\mathrm{m}}\right]. \tag{22}$$

- Insulator layer between two conductors has impedance

$$Z_i = \frac{j\omega\mu_0\mu_r^i}{2\pi}\log\left(\frac{r_o^i}{r_i^i}\right). \tag{23}$$

- Earth return impedance of the cable and mutual between cables is

$$Z_g = \frac{j\omega\,\mu_g}{2\pi}\left(-\log\left(\frac{\gamma m D}{2}\right) + \frac{1}{2} - \frac{2}{3}mH\right), \tag{24}$$

for

$$\begin{aligned} D &= \begin{cases} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} & \text{for cables } i \neq j, \\ r_i & \text{radius of the cable } i, \end{cases} \\ H &= \begin{cases} y_i + y_j & \text{for cables } i \neq j, \\ 2y_i & \text{for the cable } i, \end{cases} \end{aligned} \tag{25}$$

and $\gamma \approx 0.5772156649$ being Euler's constant.

According to [6, 7], one cable is represented with series impedance $\mathbf{Z}_{ii}$ matrix. Each matrix $\mathbf{Z}_{ii}$ has the size $n_c \times n_c$ and its components are

$$\begin{aligned} \mathbf{Z}_{ii}(j,j) &= Z_{bb}^j + Z_i^j + Z_{aa}^{j+1}, \\ \mathbf{Z}_{ii}(j,j+1) = Z_{ii}(j+1,j) &= -Z_{ab}^{j+1}, \\ \mathbf{Z}_{ii}(n_c,n_c) &= Z_{bb}^{n_c} + Z_i^{n_c} + Z_g^{ii}, \end{aligned} \tag{26}$$

and otherwise 0 for jth conductor and insulator and $j \in \{1, \dots, n_c\}$.

Interconnections with the cables are given by matrix $\mathbf{Z}_{ij}$ having all components equal Z_g^{ij}.

Shunt admittance matrix can be estimated as $\mathbf{Y} = s\mathbf{P}^{-1}$ and matrix Pm which has the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} & \cdots & \mathbf{P}_{1n} \\ \vdots & \ddots & & \vdots \\ \mathbf{P}_{n1} & \mathbf{P}_{n2} & \cdots & \mathbf{P}_{nn} \end{bmatrix}.$$

Matrices $\mathbf{P}_{ii}$ have components

$$\mathbf{P}_{ii} = \begin{bmatrix} P_c + P_s + P_a & P_s + P_a & P_a \\ P_s + P_a & P_s + P_a & P_a \\ P_a & P_a & P_a \end{bmatrix} + \begin{bmatrix} P_{ii} & P_{ii} & P_{ii} \\ P_{ii} & P_{ii} & P_{ii} \\ P_{ii} & P_{ii} & P_{ii} \end{bmatrix}, \tag{27}$$

where $P_{c,s,a}$ belong to core, shield and armor insulators and have values: $P = \frac{\log(r_o/r_i)}{2\pi\epsilon}$ and $P_{ii} = \frac{\log(2h_i/r)}{2\pi\epsilon_0}$ is a earth return. Matrices $\mathbf{P}_{ij}$, for $i \neq j$,have all components equal to $P_{ij} = \frac{\log(D_2/D_1)}{2\pi\epsilon_0}$, where $D_1 = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ and $D_2 = \sqrt{(x_i - x_j)^2 + (y_i + y_j)^2}$ [6].

As it is valid to assume that sheath and armor are grounded, it is allowed to use Kron reduction. Using Kron reduction, as proposed in [8, 9], applied to the matrices Y and Z is obtained compact shunt admittance and series impedance model. For determining ABCD parameters is used the same procedure as for the transmission line from equation (15).

2.4 AC and DC grid equivalents

AC and DC grids are represented AC and DC sources. They are described using equations:

$$\begin{aligned} \mathbf{V}_p &= \mathbf{V}_s + \mathbf{V}, \\ \mathbf{I}_p &= \mathbf{I}_s. \end{aligned} \tag{28}$$

For the impedance estimations, grid's ABCD parameters can be added as an identity matrix, because for the estimation of the equivalent impedance of the network, independent voltage sources are short connected. Additional, the internal grid impedance can be added as an serial connection of the impedance and voltage source/grid.

2.5 MMC

Voltage switching converters (VSC) are often implemented as modular multiterminal converters (MMC). They are used for the fast and efficient conversion of the energy. In this work they are represented as inverters or rectifiers, which have two DC side terminals and three AC side terminals. Each of the sides, both AC and DC, is represented with its equivalent ABCD port description. Depending on the side of interested, the corresponding ABCD representation is chosen.

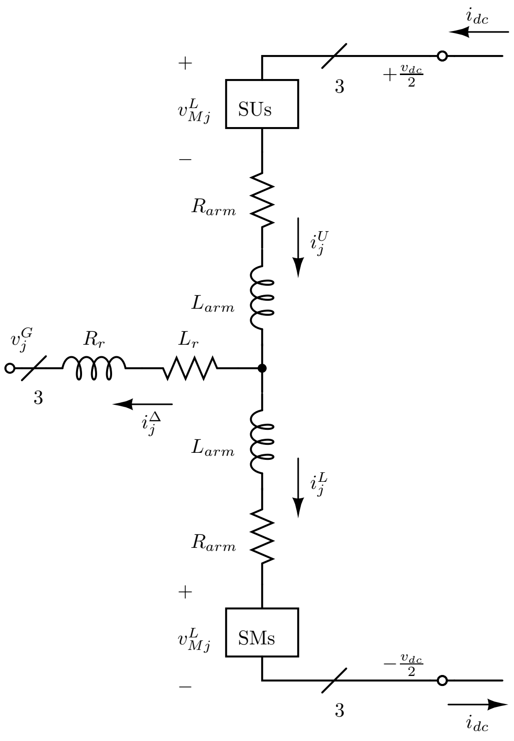


Figure 10: MMC.

One MMC is depicted in the Fig. 10. Variables from the Fig. 10 are defined for all three phases, $j \in \{a, b, c\}$. Submodules are considered with its averaged equivalent, and thus, the following equations for voltage and current can be written:

$$v_{Mj}^{U,L} = m_j^{U,L} v_{Cj}^{U,L}, \tag{29}$$

$$i_{Mj}^{U,L} = m_j^{U,L} i_j^{U,L}, \tag{30}$$

where $m_j^{U,L}$ are the corresponding insertion indices.

Using $\Sigma - \Delta$ nomenclature, the variables can be represented as:

$$\begin{aligned}
i_j^\Delta &= i_j^U - i_j^L, \quad i_j^\Sigma = \frac{i_j^U + i_j^L}{2}, \\
v_{Cj}^\Delta &= \frac{v_{Cj}^U - v_{Cj}^L}{2}, \quad v_{Cj}^\Sigma = \frac{v_{Cj}^U + v_{Cj}^L}{2}, \\
m_j^\Delta &= m_j^U - m_j^L, \quad m_j^\Sigma = m_j^U + m_j^L, \\
v_{Mj}^\Delta &= \frac{-v_{Mj}^U + v_{Mj}^L}{2} = -\frac{m_j^\Delta v_{Cj}^\Sigma + m_j^\Sigma v_{Cj}^\Delta}{2}, \\
v_{Mj}^\Sigma &= \frac{v_{Mj}^U + v_{Mj}^L}{2} = \frac{m_j^\Sigma v_{Cj}^\Sigma + m_j^\Delta v_{Cj}^\Delta}{2},
\end{aligned}$$

the equations of the state variables can be derived. In order to obtain the equations in dqz frame, Park's transformation is used in the few frames. Park's transformation is defined as:

$$P_{n\omega} = \frac{2}{3} \begin{bmatrix} \cos(n\omega t) & \cos\left(n\omega t - \frac{2\pi}{3}\right) & \cos\left(n\omega t - \frac{4\pi}{3}\right) \\ \sin(n\omega t) & \sin\left(n\omega t - \frac{2\pi}{3}\right) & \sin\left(n\omega t - \frac{4\pi}{3}\right) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}. \tag{31}$$

For the purpose of the modeling, MMC converter is represented using 12 differential equations for the state variables [10, 11]:

$$\begin{aligned}
\frac{di_d^\Delta}{dt} &= -\frac{v_d^G - v_{Md}^\Delta + R_{eq}^{ac} i_d^\Delta + \omega L_{eq}^{ac} i_q^\Delta}{L_{eq}^{ac}}, \\
\frac{di_q^\Delta}{dt} &= -\frac{v_q^G - v_{Mq}^\Delta + R_{eq}^{ac} i_q^\Delta - \omega L_{eq}^{ac} i_d^\Delta}{L_{eq}^{ac}}, \\
\frac{di_d^\Sigma}{dt} &= -\frac{v_{Md}^\Sigma + R_{arm} i_d^\Sigma - 2\omega L_{arm} i_q^\Sigma}{L_{arm}}, \\
\frac{di_q^\Sigma}{dt} &= -\frac{v_{Md}^\Sigma + R_{arm} i_q^\Sigma + 2\omega L_{arm} i_d^\Sigma}{L_{arm}}, \\
\frac{di_z^\Sigma}{dt} &= -\frac{v_{Mz}^\Sigma - \frac{V_{DC}}{2} + R_{arm} i_z^\Sigma}{L_{arm}}, \\
\frac{dv_{Cd}^\Delta}{dt} &= \frac{N}{2C_{arm}} \left(i_z^\Sigma m_d^\Delta - \frac{i_q^\Delta m_q^\Sigma}{4} + i_d^\Sigma \left(\frac{m_d^\Delta}{2} + \frac{m_{Zd}^\Delta}{2} \right) - i_q^\Sigma \left(\frac{m_q^\Delta}{2} + \frac{m_{Zq}^\Delta}{2} \right) \right. \\
&\qquad \left. i_d^\Delta \left(\frac{m_d^\Sigma}{4} + \frac{m_z^\Sigma}{2} \right) - 2\omega C_{arm}\, v_{Cq}^\Delta \right), \\
\frac{dv_{Cq}^\Delta}{dt} &= -\frac{N}{2C_{arm}} \left(\frac{i_d^\Delta m_q^\Sigma}{4} - i_z^\Sigma m_q^\Delta + i_q^\Sigma \left(\frac{m_d^\Delta}{2} - \frac{m_{Zd}^\Delta}{2} \right) + i_d^\Sigma \left(\frac{m_q^\Delta}{2} - \frac{m_{Zq}^\Delta}{2} \right) \right. \\
&\qquad \left. i_q^\Delta \left(\frac{m_d^\Sigma}{4} - \frac{m_z^\Sigma}{2} \right) - 2\omega C_{arm}\, v_{Cd}^\Delta \right), \\
\frac{dv_{CZd}^\Delta}{dt} &= -\frac{N}{8C_{arm}} \left(i_d^\Delta m_d^\Sigma + 2 i_d^\Sigma m_d^\Delta + i_q^\Delta m_q^\Sigma + 2 i_q^\Sigma m_q^\Delta + 4 i_z^\Sigma m_{Zd}^\Delta \right) - 3\omega v_{CZq}^\Delta, \\
\frac{dv_{CZq}^\Delta}{dt} &= -\frac{N}{8C_{arm}} \left(i_q^\Delta m_d^\Sigma + 2 i_d^\Sigma m_q^\Delta - i_d^\Delta m_q^\Sigma - 2 i_q^\Sigma m_d^\Delta + 4 i_z^\Sigma m_{Zq}^\Delta \right) + 3\omega v_{CZd}^\Delta, \\
\frac{dv_{Cd}^\Sigma}{dt} &= \frac{N}{2C_{arm}} \left(i_d^\Sigma m_z^\Sigma + i_z^\Sigma m_d^\Sigma + i_d^\Delta \left(\frac{m_d^\Delta}{4} + \frac{m_{Zd}^\Delta}{4} \right) - i_q^\Delta \left(\frac{m_q^\Delta}{4} - \frac{m_{Zq}^\Delta}{4} \right) \right) + 2\omega C_{arm} v_{Cq}^\Sigma, \\
\frac{dv_{Cq}^\Sigma}{dt} &= -\frac{N}{2C_{arm}} \left(i_q^\Delta \left(\frac{m_d^\Delta}{4} - \frac{m_{Zd}^\Delta}{4} \right) - i_z^\Sigma m_q^\Sigma + i_d^\Delta \left(\frac{m_q^\Delta}{4} + \frac{m_{Zq}^\Delta}{4} \right) - i_q^\Sigma m_z^\Sigma \right) + 2\omega C_{arm} v_{Cd}^\Sigma, \\
\frac{dv_{Cz}^\Sigma}{dt} &= -\frac{N}{8C_{arm}} \left(i_d^\Delta m_d^\Delta + i_q^\Delta m_q^\Delta + 2 i_d^\Sigma m_d^\Sigma + 2 i_q^\Sigma m_q^\Sigma + 4 i_z^\Sigma m_z^\Sigma \right),
\end{aligned} \tag{32}$$

where $L_{eq}^{ac} = L_f + \frac{L_{arm}}{2}$ and $R_{eq}^{ac} = R_f + \frac{R_{arm}}{2}$.

The state variables are $\mathbf{x} = [i_d^\Delta,\ i_q^\Delta,\ i_d^\Sigma,\ i_q^\Sigma,\ i_z^\Sigma,\ v_{Cd}^\Delta,\ v_{Cq}^\Delta,\ v_{CZd}^\Delta,\ v_{CZq}^\Delta,\ v_{Cd}^\Sigma,\ v_{Cq}^\Sigma,\ v_{Cz}^\Sigma]^T$.

The 12 algebraic relations used for determining 6 voltages $[v_{Md}^\Delta, v_{Mq}^\Delta, v_{MZd}^\Delta, v_{MZq}^\Delta, v_{Md}^\Sigma, v_{Mq}^\Sigma, v_{Mz}^\Sigma]$

and insertation indeices $\left[m_d^\Delta,\ \ m_q^\Delta,\ \ m_{Zd}^\Delta,\ \ m_{Zq}^\Delta,\ \ m_d^\Sigma,\ \ m_q^\Sigma,\ \ m_z^\Sigma\right]^T$ are given as:

$$\begin{aligned}
v_{Md}^\Delta &= \frac{m_q^\Delta v_{Cq}^\Sigma}{4} - \frac{m_d^\Delta v_{Cz}^\Sigma}{2} - \frac{m_d^\Delta v_{Cd}^\Sigma}{4} - \frac{m_{Zd}^\Delta v_{Cd}^\Sigma}{4} + \frac{m_{Zq}^\Delta v_{Cq}^\Sigma}{4} - \frac{m_d^\Sigma v_{Cd}^\Delta}{4} - \frac{m_z^\Sigma v_{Cd}^\Delta}{2} + \frac{m_q^\Sigma v_{Cq}^\Delta}{4} \\
&\quad - \frac{m_d^\Sigma v_{CZd}^\Delta}{4} + \frac{m_q^\Sigma v_{CZq}^\Delta}{4}, \\
v_{Mq}^\Delta &= \frac{m_d^\Delta v_{Cq}^\Sigma}{4} + \frac{m_q^\Delta v_{Cd}^\Sigma}{4} - \frac{m_q^\Delta v_{Cz}^\Sigma}{2} - \frac{m_{Zd}^\Delta v_{Cq}^\Sigma}{4} - \frac{m_{Zq}^\Delta v_{Cd}^\Sigma}{4} + \frac{m_d^\Sigma v_{Cq}^\Delta}{4} + \frac{m_q^\Sigma v_{Cd}^\Delta}{4} - \frac{m_z^\Sigma v_{Cq}^\Delta}{2} \\
&\quad - \frac{m_d^\Sigma v_{CZq}^\Delta}{4} - \frac{m_q^\Sigma v_{CZd}^\Delta}{4}, \\
v_{MZd}^\Delta &= -\frac{m_d^\Delta v_{Cd}^\Sigma}{4} - \frac{m_q^\Delta v_{Cq}^\Sigma}{4} - \frac{m_{Zd}^\Delta v_{Cz}^\Sigma}{2} - \frac{m_d^\Sigma v_{Cd}^\Delta}{4} - \frac{m_q^\Sigma v_{Cq}^\Delta}{4} - \frac{m_z^\Sigma v_{Zd}^\Delta}{2}, \\
v_{MZq}^\Delta &= -\frac{m_d^\Delta v_{Cq}^\Sigma}{4} - \frac{m_q^\Delta v_{Cd}^\Sigma}{4} - \frac{m_{Zq}^\Delta v_{Cz}^\Sigma}{2} - \frac{m_d^\Sigma v_{Cq}^\Delta}{4} + \frac{m_q^\Sigma v_{Cd}^\Delta}{4} - \frac{m_z^\Sigma v_{Zq}^\Delta}{2}, \\
v_{Md}^\Sigma &= \frac{m_d^\Delta v_{Cd}^\Delta}{4} - \frac{m_q^\Delta v_{Cq}^\Delta}{4} + \frac{m_d^\Delta v_{CZd}^\Delta}{4} + \frac{m_{Zd}^\Delta v_{Cd}^\Delta}{4} + \frac{m_q^\Delta v_{Zq}^\Delta}{4} + \frac{m_{Zq}^\Delta v_{Cq}^\Delta}{4} + \frac{m_d^\Sigma v_{Cz}^\Sigma}{2} + \frac{m_z^\Sigma v_{Cd}^\Sigma}{2}, \\
v_{Mq}^\Sigma &= \frac{m_q^\Delta v_{Zd}^\Delta}{4} - \frac{m_q^\Delta v_{Cd}^\Delta}{4} - \frac{m_d^\Delta v_{Zq}^\Delta}{4} - \frac{m_d^\Delta v_{Cq}^\Delta}{4} + \frac{m_{CZd}^\Delta v_{Cq}^\Delta}{4} - \frac{m_{Zq}^\Delta v_{Cd}^\Delta}{4} + \frac{m_q^\Sigma v_{Cz}^\Sigma}{2} + \frac{m_z^\Sigma v_{Cq}^\Sigma}{2}, \\
v_{Mz}^\Sigma &= \frac{m_d^\Delta v_{Cd}^\Delta}{4} + \frac{m_q^\Delta v_{Cq}^\Delta}{4} + \frac{m_{Zd}^\Delta v_{CZd}^\Delta}{4} + \frac{m_{Zq}^\Delta v_{CZq}^\Delta}{4} + \frac{m_d^\Sigma v_{Cd}^\Sigma}{4} + \frac{m_q^\Sigma v_{Cq}^\Sigma}{4} + \frac{m_z^\Sigma v_{Cz}^\Sigma}{2},
\end{aligned} \tag{33}$$

$$\begin{bmatrix} m_d^\Delta \\ m_q^\Delta \\ m_{Zd}^\Delta \\ m_{Zq}^\Delta \\ m_d^\Sigma \\ m_q^\Sigma \\ m_z^\Sigma \end{bmatrix} = \frac{2}{V_{DC}} \begin{bmatrix} -v_{Md,ref}^\Delta \\ -v_{Mq,ref}^\Delta \\ -v_{MZd,ref}^\Delta \\ -v_{MZq,ref}^\Delta \\ v_{Md,ref}^\Sigma \\ v_{Mq,ref}^\Sigma \\ v_{Mz,ref}^\Sigma \end{bmatrix}. \tag{34}$$

Set of the previous 12 differential equations and 12 algebraic equation is accompanied with the 6 equations for the reference values of the voltages $[v_{Md}^{\Delta *}, v_{Mq}^{\Delta *}, v_{MZd}^{\Delta *}, v_{MZq}^{\Delta *}, v_{Md}^{\Sigma *}, v_{Mq}^{\Sigma *}, v_{Mz}^{\Sigma *}]$. The reference voltages are given as zero by default, except for the value for $v_{Cz}^{\Sigma *} = V_{DC}$.

For the PI controls in the dqz frame are developed additional equations [10–12]. Several controlling methods are considered tuned using pole placement method. PLL is also implemented using PI controller [13].

- Phase locked loop (PLL)

  PLL is used to synchronize converter's frequency to the grid frequency. As all converter's variables are mapped to dqz frame using Park's transformation (31), in order to map all controls employing the frequency, following control law is formulated. According to Fig. 11 the following equations can be written.

$$\begin{aligned}
\frac{d\xi_{pll}}{dt} &= -v_q^{G,C}, \\
\frac{d\theta}{dt} &= \Delta\omega, \\
\Delta\omega &= -K_{p,pll}\ v_q^{G,C} + K_{i,pll}\ \xi_{pll}, \\
\omega_C &= \Delta\omega + \omega_0.
\end{aligned} \tag{35}$$

It should be noted output current control and circulating current control are implemented in the converter's reference frame. Thus, the rotation should be applied to the corresponding variables before the control law is applied. However, since the converter's internal dynamics is analyzed in grid's reference frame, output of the mentioned controls should be restored to the grid's reference frame using inverse of the rotation matrix. Rotation matrix is given with:

$$T(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}, \tag{36}$$

while its inverse is:

$$T^{-1}(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}. \tag{37}$$

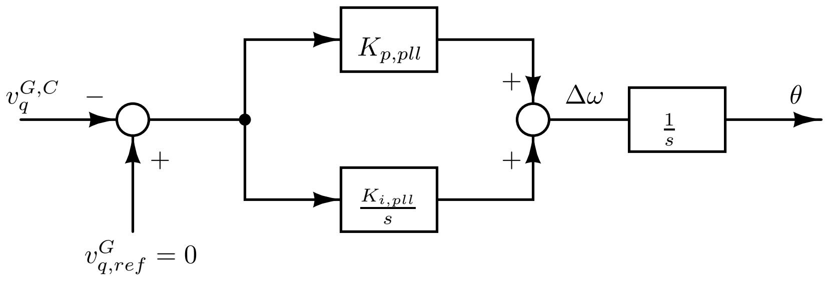


Figure 11: PLL implementation.

- Output current control

  Output current control (OCC) defines the reference values for the output currents $i_{d,ref}^{\Delta}$ and $i_{q,ref}^{\Delta}$ given in the grid reference frame. First these values have to be mapped to converter's reference frame with

$$\begin{bmatrix} i_{d,ref}^{\Delta C} \\ i_{q,ref}^{\Delta C} \end{bmatrix} = T(\theta) \begin{bmatrix} i_{d,ref}^{\Delta} \\ i_{q,ref}^{\Delta} \end{bmatrix},$$

  and currents i_d^{Δ} and i_q^{Δ} are also mapped to:

$$\begin{bmatrix} i_d^{\Delta C} \\ i_q^{\Delta C} \end{bmatrix} = T(\theta) \begin{bmatrix} i_d^{\Delta} \\ i_q^{\Delta} \end{bmatrix}.$$

  Then, this control method adds several equations:

$$\begin{aligned} \frac{d\xi_d^{\Delta}}{dt} &= i_{d,ref}^{\Delta C} - i_d^{\Delta C}, \\ \frac{d\xi_q^{\Delta}}{dt} &= i_{q,ref}^{\Delta C} - i_q^{\Delta C}, \\ v_{Md,ref}^{\Delta C} &= K_{i,occ}\xi_d^{\Delta} + K_{p,occ}\,(i_{d,ref}^{\Delta C} - i_d^{\Delta C}) + \omega_C L_{eq}^{ac} i_q^{\Delta} + v_d^{G,C}, \\ v_{Mq,ref}^{\Delta C} &= K_{i,occ}\xi_q^{\Delta} + K_{p,occ}\,(i_{q,ref}^{\Delta C} - i_q^{\Delta C}) - \omega_C L_{eq}^{ac} i_d^{\Delta C} + v_q^{G,C}. \end{aligned} \tag{38}$$

  Voltages $v_{Md,ref}^{\Delta C}$ and $v_{Mq,ref}^{\Delta C}$ are used in grid's reference frame for the further calculations:

$$\begin{bmatrix} v_{Md,ref}^{\Delta} \\ v_{Mq,ref}^{\Delta} \end{bmatrix} = T^{-1}(\theta) \begin{bmatrix} v_{Md,ref}^{\Delta C} \\ v_{Mq,ref}^{\Delta C} \end{bmatrix}. \tag{39}$$

  If the controller is defined only using bandwidth ω_n and ζ, the proportional and integral gains are then tuned as:

$$\begin{aligned} K_{i,occ} &= L_{eq}^{ac}\,\omega_n^2, \\ K_{p,occ} &= 2\zeta\omega_n L_{eq}^{ac} - R_{eq}^{ac}. \end{aligned}$$

In case that the reference for the output currents is not provided, it is assumed to be: $i^{\Delta}_{d,ref} = -\frac{2S_{base}}{3V_{base}\sqrt{\frac{2}{3}}}$ and $i^{\Delta}_{q,ref} = 0$.

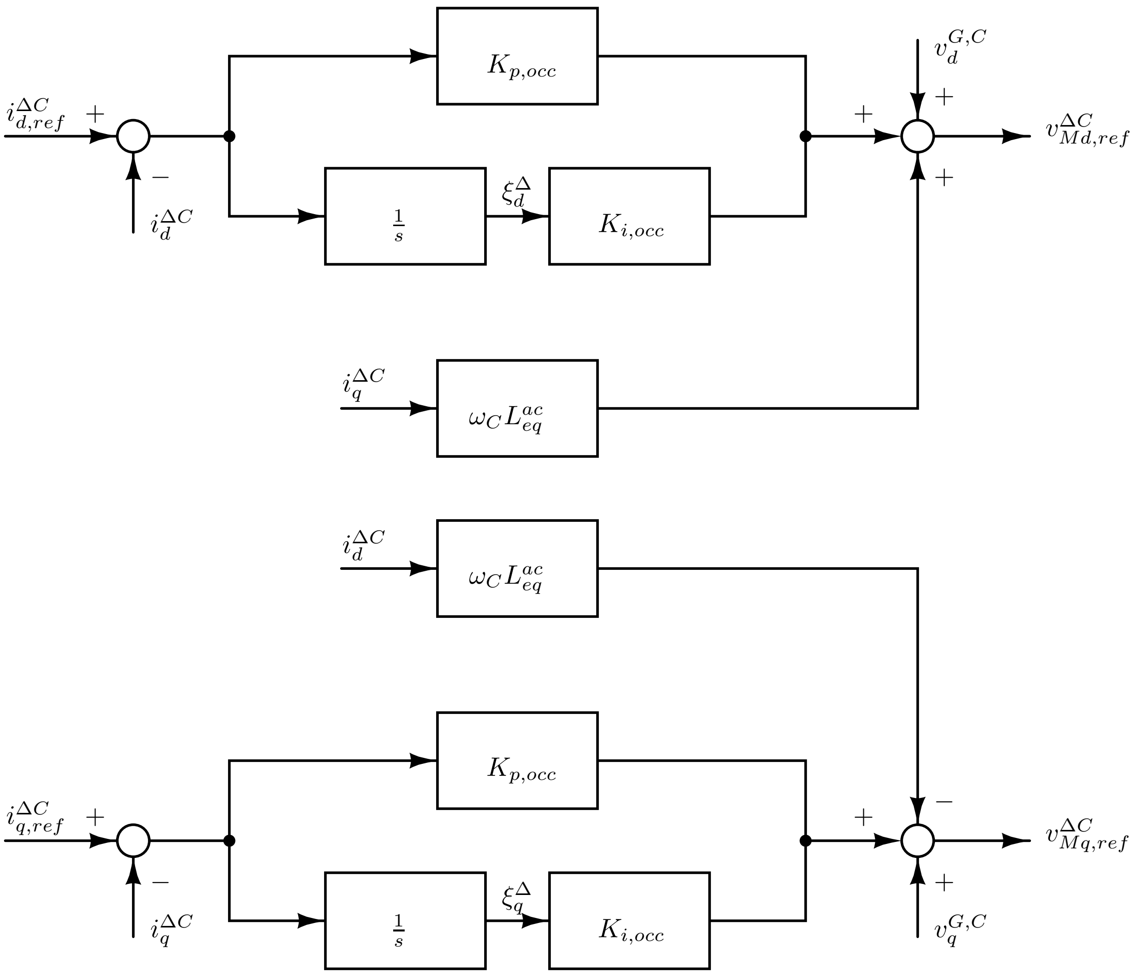


Figure 12: OCC implementation.

- Circulating current control

  Similarly as OCC, circulating current control (CCC) is constructed to set the circulating current to it reference, which is considered to be $i^{\Sigma}_{d,ref} = 0$, $i^{\Sigma}_{q,ref} = 0$. Control is implemented in the converter's reference frame, and thus:

$$\begin{bmatrix} i^{\Sigma C}_{d,ref} \\ i^{\Sigma C}_{q,ref} \end{bmatrix} = T(-2\theta) \begin{bmatrix} i^{\Sigma}_{d,ref} \\ i^{\Sigma}_{q,ref} \end{bmatrix}, \qquad \begin{bmatrix} i^{\Sigma C}_{d} \\ i^{\Sigma C}_{q} \end{bmatrix} = T(-2\theta) \begin{bmatrix} i^{\Sigma}_{d} \\ i^{\Sigma}_{q} \end{bmatrix}.$$

  The equations added by CCC are:

$$\begin{aligned} \frac{d\xi^{\Sigma}_{d}}{dt} &= i^{\Sigma C}_{d,ref} - i^{\Sigma C}_{d}, \\ \frac{di^{\Sigma}_{q}}{dt} &= i^{\Sigma C}_{q,ref} - i^{\Sigma C}_{q}, \\ v^{\Sigma C}_{Md,ref} &= -K_{i,ccc}\,\xi^{\Sigma}_{d} - K_{p,ccc}\,(i^{\Sigma C}_{d,ref} - i^{\Sigma C}_{d}) + 2\omega_C L_{arm} i^{\Sigma C}_{q}, \\ v^{\Sigma C}_{Mq,ref} &= -K_{i,ccc}\xi^{\Sigma}_{q} - K_{p,ccc}\,(i^{\Sigma C}_{q,ref} - i^{\Sigma C}_{q}) - 2\omega_C L_{arm} i^{\Sigma C}_{d}. \end{aligned} \tag{40}$$

  To return to grid's reference frame, the following is applied:

$$\begin{bmatrix} v^{\Sigma}_{Md,ref} \\ v^{\Sigma}_{Mq,ref} \end{bmatrix} = T^{-1}(-2\theta) \begin{bmatrix} v^{\Sigma C}_{Md,ref} \\ v^{\Sigma C}_{Mq,ref} \end{bmatrix}. \tag{41}$$

The proportional and integral gains are tuned:

$$\begin{aligned} K_i &= L_{arm}\,\omega_n^2, \\ K_p &= 2\zeta\omega_n L_{arm} - R_{arm}. \end{aligned}$$

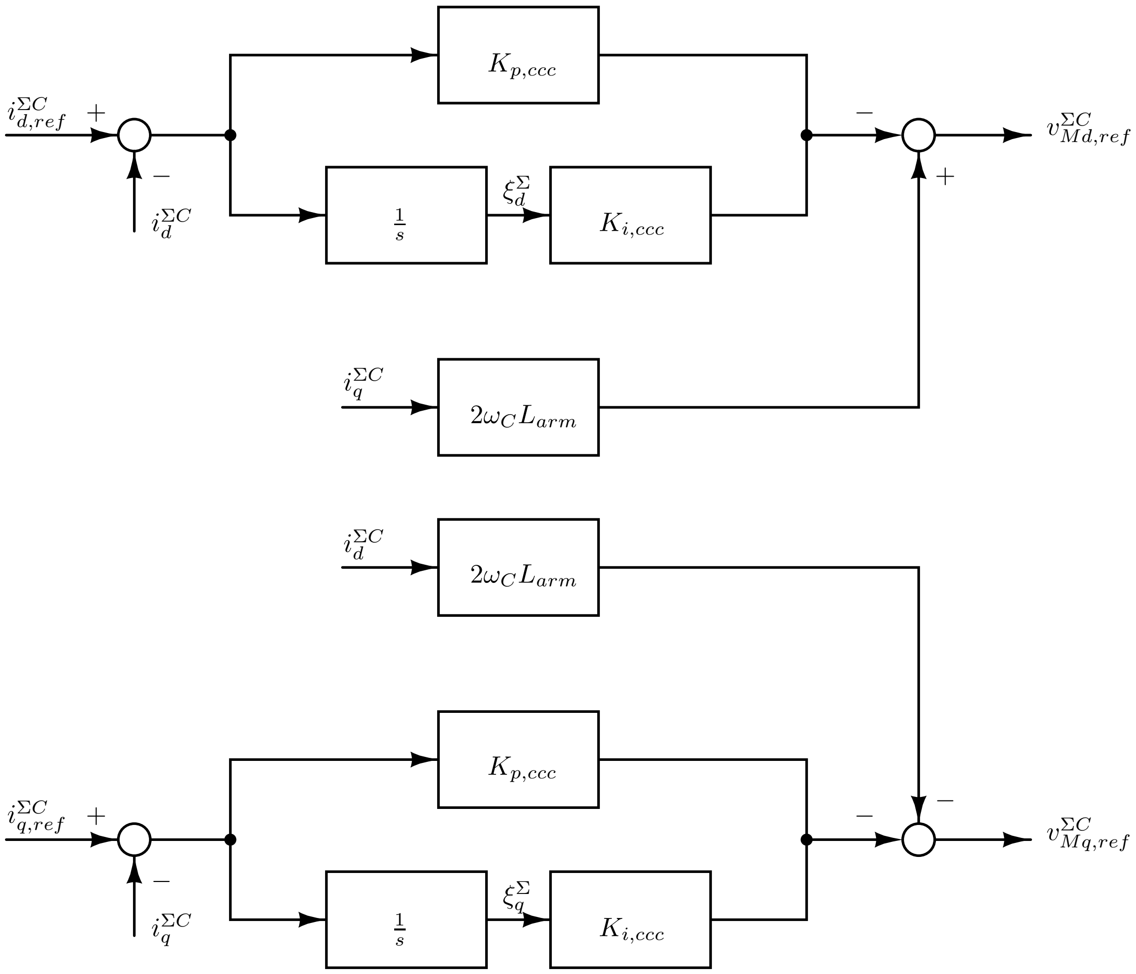


Figure 13: CCC implementation.

- Energy control and zero current control

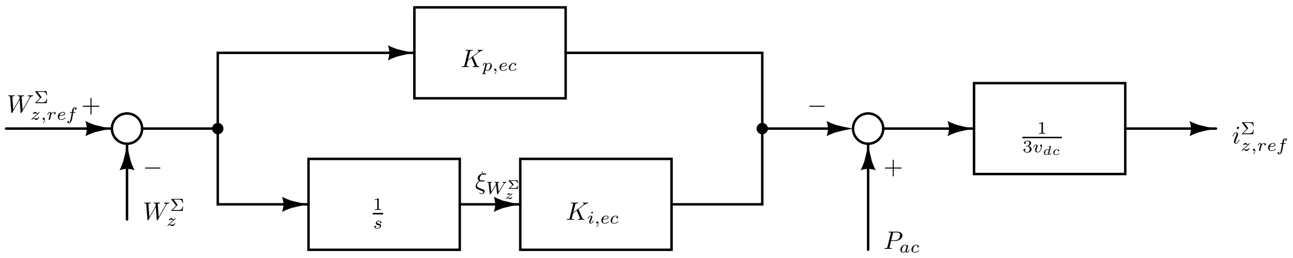
  Energy control is built around the "zero" energy and as a result it provides a reference value for the "zero" current $i^{\Sigma}_{z,ref}$. Energy controller involves the following equations as visible from Fig. 14a.

$$\begin{aligned} W^{\Sigma}_{z} &= \frac{3C_{arm}}{2N}\,({v^{\Delta}_{Cd}}^2 + {v^{\Delta}_{Cq}}^2 + {v^{\Delta}_{CZd}}^2 + {v^{\Delta}_{CZq}}^2 + {v^{\Sigma}_{Cd}}^2 + {v^{\Sigma}_{Cq}}^2 + 2{v^{\Sigma}_{Cz}}^2), \\ \frac{d\xi_{W^{\Sigma}_{z}}}{dt} &= W^{\Sigma}_{z,ref} - W^{\Sigma}_{z}, \\ P_{ac} &= \frac{3}{2}\,(v^{G,C}_{d} i^{\Delta C}_{d} + v^{G,C}_{q} i^{\Delta C}_{q}), \\ i^{\Sigma}_{z,ref} &= \frac{K_{p,ec}\,(W^{\Sigma}_{z,ref} - W^{\Sigma}_{z}) + K_{i,ec}\xi_{W^{\Sigma}_{z}} + P_{ac}}{3v_{dc}}. \end{aligned} \tag{42}$$

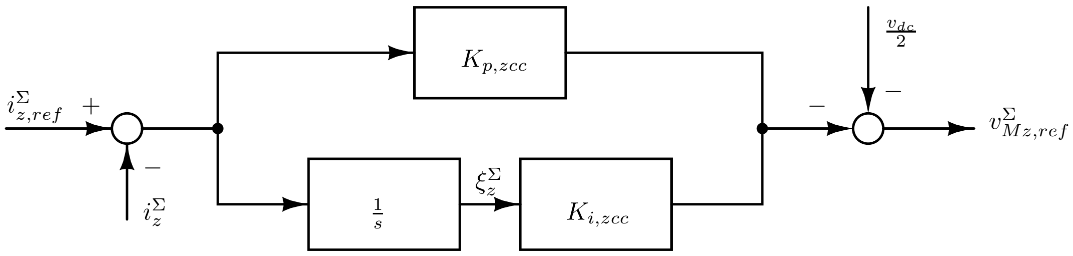
  Additionally, the zero current control (ZCC) sets the zero current to the desired value. The implementation of this control is depicted in Fig. 14b. It can work without the energy controller.

$$\begin{aligned}\frac{d\xi_z^\Sigma}{dt} &= i_{z,ref}^\Sigma - i_z^\Sigma,\\ v_{Mz,ref}^\Sigma &= \frac{v_{dc}}{2} - K_{p,zcc}\,(i_{z,ref}^\Sigma - i_z^\Sigma) - K_{i,zcc}\,\xi_z^\Sigma.\end{aligned} \tag{43}$$

Tuning of the ZCC is the same as for CCC.



(a)



(b)

Figure 14: Energy control and ZCC implementation.

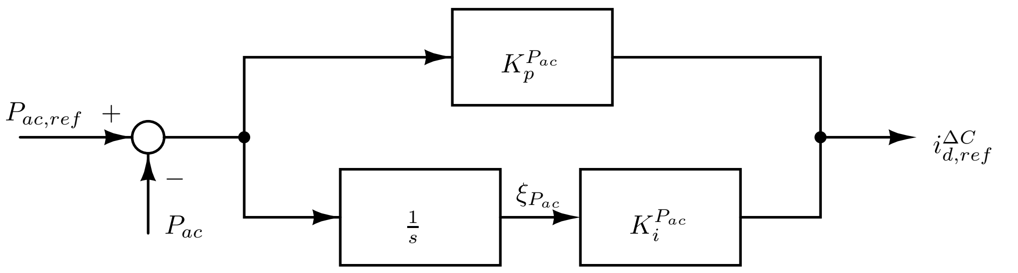
- Active and reactive power control

  An outer control loop for the control of the active and reactive power can be added, see Fig. 15. These control loops are used to successfully estimate AC currents $i_{dq,ref}^\Delta$. The control loops operate according to equations:

$$\begin{aligned}P_{ac} &= \frac{3}{2}\,(v_d^{G,C} i_d^{\Delta C} + v_q^{G,C} i_q^{\Delta C}),\\ Q_{ac} &= \frac{3}{2}\,(-v_d^{G,C} i_q^{\Delta C} + v_q^{G,C} i_d^{\Delta C}),\\ \frac{d\xi_{P_{ac}}}{dt} &= P_{ac,ref} - P_{ac},\\ \frac{d\xi_{Q_{ac}}}{dt} &= Q_{ac,ref} - Q_{ac},\\ i_{d,ref}^{\Delta C} &= K_p^{P_{ac}}\,(P_{ac,ref} - P_{ac}) + K_i^{P_{ac}}\xi_{P_{ac}},\\ i_{q,ref}^{\Delta C} &= -K_p^{Q_{ac}}\,(Q_{ac,ref} - Q_{ac}) - K_i^{Q_{ac}}\xi_{Q_{ac}}.\end{aligned} \tag{44}$$

  If not specified differently, the reference value for the active power is $-S_{base}$ and for the reactive power is zero.

The previous system of differential and algebraic equations is solved for the equilibrium using Julia package NLsolve. After determining of the equilibrium, the system is represented as a multi-input multi-output system (MIMO), where the variables



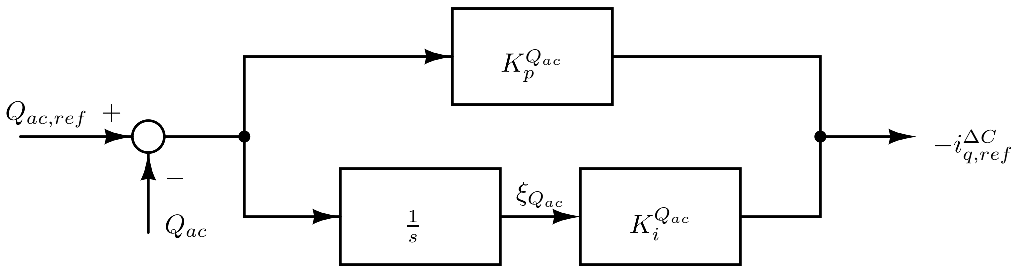


Figure 15: Active and reactive power control implementation.

$\mathbf{x} = \begin{bmatrix} i_d^\Delta & i_q^\Delta & i_d^\Sigma & i_q^\Sigma & i_z^\Sigma & v_{Cd}^\Delta & v_{Cq}^\Delta & v_{CZd}^\Delta & v_{CZq}^\Delta & v_{Cd}^\Sigma & v_{Cq}^\Sigma & v_{Cz}^\Sigma \end{bmatrix}$ represent the state-variables and input vector is given as $\mathbf{u} = \begin{bmatrix} v_{dc} & v_d^G & v_q^G \end{bmatrix}$. In order to get transfer functions from the input to output given with the values $\mathbf{y} = \begin{bmatrix} i_d^\Delta & i_q^\Delta & i_z^\Sigma \end{bmatrix}$, the previous equations are rewritten to satisfy the following form:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}_{MIMO}\mathbf{x}(t) + \mathbf{B}_{MIMO}\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C}_{MIMO}\mathbf{x}(t) + \mathbf{D}_{MIMO}\mathbf{u}(t). \end{aligned} \tag{45}$$

Corresponding matrices $\mathbf{A}_{MIMO}$, $\mathbf{B}_{MIMO}$, $\mathbf{C}_{MIMO}$ and $\mathbf{D}_{MIMO}$ are determined as Jacobians around the equilibrium for the state variables and inputs. Jacobian is successfully determined using Julia package ForwardDiff [14]. Applying the Laplace transform, the previous system of equations (45) transforms to:

$$\begin{aligned} s\mathbf{X}(s) &= \mathbf{A}_{MIMO}\mathbf{X}(s) + \mathbf{B}_{MIMO}\mathbf{U}(s), \\ \mathbf{Y}(s) &= \mathbf{C}_{MIMO}\mathbf{X}(s) + \mathbf{D}_{MIMO}\mathbf{U}(s). \end{aligned} \tag{46}$$

The MIMO transfer function is thus given by:

$$\mathbf{Y}_{MMC}(s) = \mathbf{Y}(s)\mathbf{U}(s)^{-1} = \mathbf{C}_{MIMO}\left(s\mathbf{I} - \mathbf{A}_{MIMO}\right)^{-1}\mathbf{B}_{MIMO} + \mathbf{D}_{MIMO}. \tag{47}$$

Obtained matrix transfer function presents admittances

$$\mathbf{Y}_{MMC}(s) = \begin{bmatrix} Y_{dz} & Y_{dd} & Y_{dq} \\ Y_{qz} & Y_{qd} & Y_{qq} \\ Y_{zz} & Y_{zd} & Y_{zq} \end{bmatrix} \tag{48}$$

that connect vector of currents $[i_d^\Delta(s),\ i_q^\Delta(s),\ i_z^\Sigma(s)]^T$ with the voltages vector $[v_{dc}(s),\ v_d^G(s),\ v_q^G(s)]^T$.

If we model the converter as in Fig. 16, then MMC can be represented as one input, two output component.

As ABCD parameters work correctly for the components with the same number of the input and the output pins, the equations which are solved for the MMC are given with the Y_{MMC} matrix as it follows:

$$\begin{bmatrix} i_{dc} \\ i_d^\Delta \\ i_q^\Delta \end{bmatrix} = \begin{bmatrix} 3Y_{zz} & 3Y_{zd} & 3Y_{zq} \\ Y_{dz} & Y_{dd} & Y_{dq} \\ Y_{qz} & Y_{qd} & Y_{qq} \end{bmatrix} \begin{bmatrix} v_{dc} \\ v_d^G \\ v_q^G \end{bmatrix}. \tag{49}$$

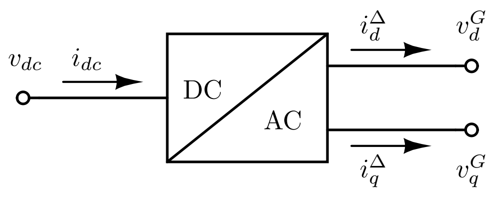


Figure 16: MMC block model.

3 Power flow

The designed network is initialized using optimal power flow tool [15] implemented as a Julia package, which can be found in the repository [16]. The package relies on the power flow models developed for the MatACDC simulator [17], which extends Matpower [18] AC power system models with the DC representations and with power converters.

As a results, the constructed power system contains AC and DC branches and buses, converters, generators, loads, shunts and storage elements. Components implemented in this simulator are represented using their equivalent models for the purpose of the power flow analysis.

3.1 AC and DC branches

AC and DC branches represent three-phase and DC connections between buses, respectively. Branches are grouped inside AC or DC grids (zones). AC branches are defined with parameters described in [18], while DC branches parameters are given in [17].

For the purpose of the system components modeling, in Fig. 17 is depicted the model of the AC branch as provided in [18]. Beside the shunt admittance $j\frac{b_c}{2}$, Julia package [16] supports the admittance real part as $\frac{g_c}{2} + j\frac{b_c}{2}$. The full expression for the AC admittance parameters is given by the equation:

$$\mathbf{Y}_{ac} = \begin{bmatrix} \left(y_s + \frac{g_c}{2} + j\frac{b_c}{2}\right)\frac{1}{\tau^2} & -\frac{y_s}{\tau \exp(-j\theta_{\text{shift}})} \\ -\frac{y_s}{\tau \exp(-j\theta_{\text{shift}})} & \left(y_s + \frac{g_c}{2} + j\frac{b_c}{2}\right) \end{bmatrix}. \tag{50}$$

It should be noted that the AC network is considered as balanced, and thus, all the components have the matrix models with only diagonal same valued elements.

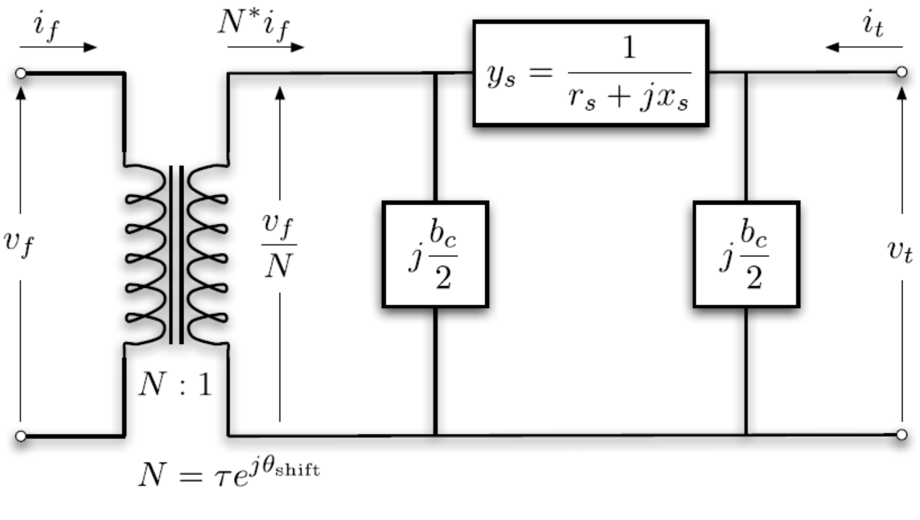


Figure 17: Matpower AC branch model [18].

DC branch is modeled with its equivalent series resistance [17].

Some power flow components are in the power flow simulation modeled as AC and DC branches and their models will be desribed in detail in this subsection.

3.1.1 Impedance

As described in the subsection 2.1, impedance is modeled using ABCD parameters:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{Z} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}. \tag{51}$$

In the case of the DC impedance, all matrices are of the size 1×1, while in the three-phase case they are 3×3.

DC branch model is then given as $r = \Re\{Z\}$. AC branch model is according to its definition given as an ideal transformer with $\tau = 1$ and $\theta_{\text{shift}} = 0$, and with $r_s = \Re\{\mathbf{Z}(j\omega)\langle 1,1\rangle\}$, $x_s = \Im\{\mathbf{Z}(j\omega)\langle 1,1\rangle\}$, $g_c = 0$ and $b_c = 0$.

3.1.2 Transformer

Since the transformer model considered in 2.2 cannot be easily represented as the model in the Fig. 17, from the obtained ABCD parameters are first constructed Y parameters.

$$\mathbf{Y} = \begin{bmatrix} \mathbf{D}\mathbf{B}^{-1} & \mathbf{C} - \mathbf{D}\mathbf{B}^{-1}\mathbf{A} \\ -\mathbf{B}^{-1} & \mathbf{B}^{-1}\mathbf{A} \end{bmatrix}. \tag{52}$$

For the case of the Dc branch, since ABCD parameters are each 1×1 matrices or scalars, it can be easily estimated tap value as $\tau = \sqrt{\frac{A}{D}}$. Series impedance that represents DC branch is then $r = \Re\{\frac{B}{\tau}\}$.

For the case of the AC network, using the assumption of the balanced system, the submatrices $\mathbf{Y}\langle 1:3, 1:3\rangle$, $\mathbf{Y}\langle 1:3, 4:6\rangle$, $\mathbf{Y}\langle 4:6, 1:3\rangle$ and $\mathbf{Y}\langle 4:6, 4:6\rangle$ are diagonal. Thus, it is sufficient to use a single diagonal value from each submatrix. Then, $\mathbf{Y}\langle 1,1\rangle = \left(y_s + \frac{g_c}{2} + j\frac{b_c}{2}\right)\frac{1}{\tau^2}$, $\mathbf{Y}\langle 1,4\rangle = \mathbf{Y}\langle 4,1\rangle = -\frac{y_s}{\tau\exp(-j\theta_{\text{shift}})}$ and $\mathbf{Y}\langle 4,4\rangle = \left(y_s + \frac{g_c}{2} + j\frac{b_c}{2}\right)$. The following expressions are derived:

$$\begin{gathered}
\tau = \sqrt{\frac{\mathbf{Y}\langle 4,4\rangle}{\mathbf{Y}\langle 1,1\rangle}}, \quad \theta_{\text{shift}} = 0, \\
y_s = -\mathbf{Y}\langle 1,4\rangle\,\tau\,\exp(-j\theta_{\text{shift}}), \\
y_c = \mathbf{Y}\langle 4,4\rangle - y_s, \\
r_s = \Re\left\{\frac{1}{y_s}\right\}, \quad x_s = \Im\left\{\frac{1}{y_s}\right\}, \\
g_c = \Re\{y_c\}, \quad b_c = \Im\{y_c\}.
\end{gathered}$$

3.1.3 Transmission line

Transmission line is represented using its nominal π-model depicted in Fig. 18, where

$$\begin{aligned}
\mathbf{Z}(j\omega) &= \mathbf{Y}_c^{-1}\sinh(\mathbf{\Gamma}l), \\
\mathbf{Y}(j\omega) &= \mathbf{Y}_c\tanh(\mathbf{\Gamma}l).
\end{aligned} \tag{53}$$

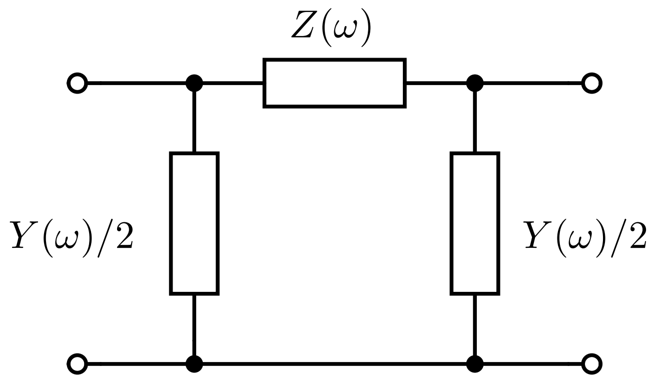


Figure 18: Nominal π-model of the transmission line.

For the DC case, the shunt admittance is not considered, while the branch resistance is equal to $r = \Re\{Z(0)\}$.

For the balanced AC transmission line, the impedance and admittance matrices are diagonal. It can be chosen $Z(j\omega) = \mathbf{Z}(j\omega)\langle 1,1\rangle$ and $Y(j\omega) = \mathbf{Y}\langle 1,1\rangle$. Then, the AC branch model is given with:

$$\begin{gathered} \tau = 0, \quad \theta_{\text{shift}} = 0, \\ r_s = \Re\{Z(j\omega)\}, \quad x_s = \Im\{Z(j\omega)\}, \\ g_c = \Re\{Y(j\omega)\}, \quad b_c = \Im\{Y(j\omega)\}. \end{gathered} \tag{54}$$

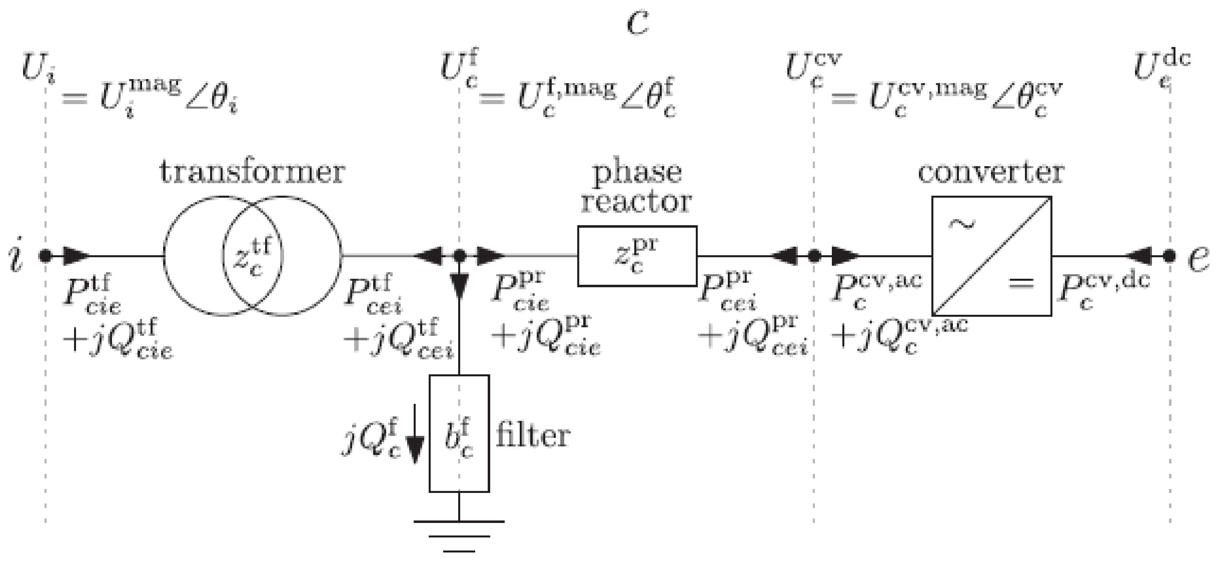
3.2 Generators

In this simulator three-phase AC source presents a generator. Generators are defined as reference buses.

3.3 Power converter

Power converter is according to [17] modeled together with the reactor, filter and a transformer. In order to fit the constructed MMC model in the section 2.5, only the reactor is considered.

Losses of the converter are calculated in the form of $P_{loss} = a + bI_c + cI_c^2$. Since the switches are modeled as ideal, our model supports only constant $c = \frac{R_{arm}}{2}$.



c
$U_i = U_i^{\text{mag}}\angle\theta_i$
$U_c^{\text{f}} = U_c^{\text{f,mag}}\angle\theta_c^{\text{f}}$
$U_c^{\text{cv}} = U_c^{\text{cv,mag}}\angle\theta_c^{\text{cv}}$
U_e^{dc}
transformer
phase reactor
converter
i
z_c^{tf}
z_c^{pr}
e
$P_{cie}^{\text{tf}} + jQ_{cie}^{\text{tf}}$
$P_{cei}^{\text{tf}} + jQ_{cei}^{\text{tf}}$
$P_{cie}^{\text{pr}} + jQ_{cie}^{\text{pr}}$
$P_{cei}^{\text{pr}} + jQ_{cei}^{\text{pr}}$
$P_c^{\text{cv,ac}} + jQ_c^{\text{cv,ac}}$
$P_c^{\text{cv,dc}}$
jQ_c^{f}
b_c^{f}
filter

Figure 19: Power flow model of the power converter.

4 Simulator impementation

The simulator is implemented in Julia programming language and it consists of multiple structures with the different priority. The highest priority has a structure named `Network`, which is "responsible" for the creation of the complete system. Structure `Network` consists of the `Nets` and `Elements`. Each `Net` is a group of connection between elements, and group `Elements` presents a collection of all the elements in the `Network` (system).

`Element` can be made as a special power system component, or as a composite element, which can contain a specific number of components and their connection. The composite element is by default made of one network.

Simulator is implemented using the following Julia packages: SymEngine, LinearAlgebra, NLsolve and ForwardDiff [14] for symbolic and numerical calculations; DelimittedFiles and FileIO for the reading and writing in file; DataStructures and Parameters for data types support; Plots, LaTeXStrings and DSP for the visualisation.

4.1 Network structure

Network presents a struct with the collection of all components and their interconnections. Components are defined as type Element and the nodes present the dictionary, in which each node has its Symbol and it consists of the element pins connected to it.

Network is initialized with the macro network as it follows.

```
@network begin #= ... =# end
```

Provides a simple domain-specific language to decribe networks. The begin/end block can hold element definitions of the form `refdes = elementfunc(params)` and connection specifications of the following form. Instead of ⟷, it can be used ==.

```
refdes[pin1] ⟷ refdes2[pin2]
```

Examples:

To construct the network with one impedance (resistor) and one voltage source use the following code.

```
@network begin
    src = dc_source(voltage = 5)
    r = impedance(z = 1000, pins = 1)
    src[1.1] ⟷ r[1.1]
    src[2.1] ⟷ r[2.1]
end
```

Alternatively, connection specifications can be given after an element specification, separated by commas. In that case, the refdes may be omitted, defaulting to the current element.

```
@network begin
    src = dc_source(voltage = 5)
    r = impedance(z = 1000, pins = 1), src[1.1] ⟷ [1.1], src[2.1] ⟷ [2.1]
end
```

Finally, a connection endpoint may simply be of the form `netname`, to connect to a named net. (Such named nets are created as needed.)

```
@network begin
    src = dc_source(voltage = 5), [2.1] ⟷ gnd
    r = impedance(z = 1000, pins = 1), [1.1] ⟷ src[1.1], [2.1] ⟷ gnd
end
```

4.1.1 Add and delete components

Different power system components are added as type `Element` in the `Network`. Functions that add and delete components are as follows.

- `add!(n::Network, elem::Element)`

  Adds the element `elem` to the network `n`, creating and returning a new, unique reference designator, leaving its pins unconnected.

- `add!(n::Network, designator::Symbol, elem::Element)`

  Adds the element `elem` to the network `n` with the reference designator `designator`, leaving its pins unconnected. If the network already contained an element named designator, it is removed first.

- `delete!(n::Network, designator::Symbol)`

  Deletes the element named `designator` from the network `n` (disconnecting all its pins).

- `connect!(n::Network, pins::UnionSymbol,TupleSymbol,Any...)`

  Connects the given pins (or named nets) to each other in the network `n`. Named nets are given as `Symbols`, pins are given as `Tuple{Symbols,Any}`, where the first entry is the reference `designator` of an element in `n`, and the second entry is the pin name. For convenience, the latter is automatically converted to a `Symbol` as needed.

- `disconnect!(n::Network, p::TupleSymbol,Symbol)`

  Disconnects the given pin `p` from anything else in the network `n`. The pin is given as a `Tuple{Symbols,Any}`, where the first entry is the reference `designator` of an element in `n`, and the second entry is the pin name. For convenience, the latter is automatically converted to a `Symbol` as needed. Note that if e.g. three pin `p1`, `p2`, and `p3` are connected then `disconnect!(n, p1)` will disconnect `p1` from `p2` and `p3`, but leave `p2` and `p3` connected to each other.

- `composite_element(net, pinmap)`

  Create a net element from the (sub-)network net. The pinmap defines the mapping from pin names of the element to be created to pins (or nets) in net. Optionally, ports can be used to explicitly specify (as a list of pairs of pins) the ports to use. By default, a port will be created between one arbitrarily chosen pin and every other pin.

  Note that the pins still implicitly specify ports, so an even number must be given, but the same pin may be given multiple times to be part of multiple ports.

Example: Using functions `add!` and `connect!`.

```
network = Network()
add!(network, :r, impedance(z = 1e3, pins = 1))
add!(network, :src, dc_source(voltage = 5))
connect!(network, (:src, 2.1), (:r, 2.1), :gnd) # connect to gnd net
```

Example: Creating element from the network.

```
net = @network begin
   r1 = impedance(z = 10e3, pins = 1)
   r2 = impedance(z = 10e3, pins = 1), [1.1] == r1[2.1]
   c = impedance(z = 10e3, pins = 1), [1.1] == r2[1.1], [2.1] == r2[2.1]
   src = dc_source(voltage = 5), [1.1] == r1[1.1], [2.1] == r2[2.1]
end
composite_element(net, Any[(:r2, Symbol(1.1)), (:r2, Symbol(2.1))])
```

4.1.2 Additional checks

After network `net` construction, at the end of macro `net = @network begin #= ... =# end`, the program calls following two functions to check if the networks is well connected and to construct ABCD parameter equivalents for all the components. The following functions are only called internally.

- `check_lumped_elements(net :: Network)`

  This function checks if the network is well connected, e.g. that there is no lumped (disconnected) pins in the network.

- `power_flow(net :: Network)`

  This function is called in order to set converters' operating points. It generates a dictionary with the syntax used in [16] and it solves power flow problem using the Julia package PowerModelsACDC [16]. Results are used to update operating point of the models.

4.1.3 Determine impedance and stability of the port

For the selected multiport we can determine impedance using the procedure described in section 1. Multiport is depicted in Fig. 1. The following function generates "visible" impedance from the port defined by input and output pins. Impedance is determined in the selected number of points, which is by default 1000.

Port pins can possibly be connected to some elements, which should not be considered for the impedance estimation. Those elements are listed as `elim_elements`.

```
function determine_impedance(network::Network; input_pins :: ArrayAny,
    output_pins :: ArrayAny, elim_elements :: ArraySymbol,
    omega_range = (-3, 5, 100))
```

Example: Specification for the impedance estimation is given on the example of the following network that consists of the DC voltage source and a cable.

```
net = @network begin
    vs = dc_source(voltage = 500e3)
    c = cable(length = 100e3, positions = [(0,1)], earth_parameters = (1,1,1),
    C1 = Conductor(r_o = 24.25e-3, ρ = 1.72e-8),
    C2 = Conductor(r_i = 41.75e-3, r_o = 46.25e-3, ρ = 22e-8),
    C3 = Conductor(r_i = 49.75e-3, r_o = 60.55e-3, ρ = 18e-8, μ_r = 10),
    I1 = Insulator(r_i = 24.25e-3, r_o = 41.75e-3, ϵ_r = 2.3),
    I2 = Insulator(r_i = 46.25e-3, r_o = 49.75e-3, ϵ_r = 2.3),
    I3 = Insulator(r_i = 60.55e-3, r_o = 65.75e-3, ϵ_r = 2.3))
    vs[1.1] ⟷ c[1.1] ⟷ Node1
    vs[2.1] ⟷ c[2.1] ⟷ gnd
end
```

To determine impedance visible from the voltage source `vs`, the following command should be called:

```
imp, omega = determine_impedance(net, elim_elements = [:vs],
        input_pins = Any[:Node1], output_pins = Any[:gnd],
        omega_range = (-1,6,10000))
```

The impedance is determined inside network `net`, from the element `vs` (without including the element `vs`) and the port defined with `input_pins` as array consisting of `Node1` and the `output_pins` containing array with `gnd`. Impedance is estimated for the frequency in [rad/s] with the range 10^{-1} to 10^{6} in 10000 points.

The function returns the two complex arrays: first is the impedance array and the second one is frequency array.

4.1.4 Saving and plotting data

Any component in the network has its own data representation. Component data can be saved and plotted using the following functions.

- `save_data(element :: Element, file_name :: String;`
  `omega_range = (-3, 5, 1000), scale = :log)`

  This function saves component specific data in csv textual file. The data is saved as frequency dependent. Thus, an additional parameter `omega_range` provides possibility to manually add the frequency range for saving data. The scale can be given as logarithmic with `:log` and linear with `:lin`.

- `plot_data(element :: Element; omega_range = (-3, 5, 1000), scale = :log)`

  Component defined function for plotting data. It differs from component to component. It plots the component defined data with the desired frequency range. Additional parameter `omega_range` provides possibility to manually add the frequency scale for saving data. Scale can be given as logarithmic with `:log` and linear with `:lin`.

- `bode(impedance :: Array{Any}; omega_range = (-3, 5, 100),`
  `titles :: Array{String} = [""], omega = [])`

  Used for plotting determined impedance or a frequency dependent data as a bode plot. Function takes frequency points in the form of `omega_range` or as mapped values `omega`. For a nice diagrams, the labels can be given as `titles`.

4.2 Component definition

4.2.1 Impedance

Impedance is constructed as an element by calling the function: `impedance(;z :: Union{Int, Float64, Basic, Array{Basic}} = 0, pins :: Int = 0)`.

The function creates impedance with specified number of input/output pins. The number of input pins is equal to the number of output pins. The impedance expression `exp` has to be given in ohms and can have both numerical and symbolic value (example: `z = s-2`). Pins are named: 1.1, 1.2, ..., 1.n and 2.1, 2.2, ..., 2.n, where n is a number of input/output pins.

In the case of 1×1 impedance, parameter `z` has only one value. Example: `impedance(z = 1000, pins = 1)`.

If the impedance is multiport, then its value is given as an array `[val]` with one, n or $n \times n$ number of values. In the case of one element, then the impedance has only diagonal nonzero values equal to val. In the case of pins number of values, the impedance has only diagonal nonzero values equal to the values in an array val. If the array has the size $n \times n$, impedance matrix has the size $n \times n$ and all its values are defined.

Examples:

```
# 3×3 impedance with diagonal values equal to s
impedance(z = [s], pins = 3)
# 3×3 impedance with diagonal values equal 2, s, 0.5s, respectively
impedance(z = [2,s,s/2], pins = 3)
# 2×2 impedance with all values defined
impedance(z = [1,s,3,4], pins = 2)
```

In this example with `s` is specified Laplace parameter. The impedance can be added in the form of a transfer function (both linear and nonlinear).

Example of the network with impedances, depicted in Fig. 20, and its bode plot.

```
using SymEngine

s = symbols("s")
net = @network begin
    vs = dc_source(voltage = 5)
    z1 = impedance(z = s+2, pins = 1)
    z2 = impedance(z = s, pins = 1)
    z3 = impedance(z = s, pins = 1)

    vs[1.1] ⟷ z1[1.1] ⟷ Node1
    z1[2.1] ⟷ z2[1.1] ⟷ z3[1.1]
    vs[2.1] ⟷ z2[2.1] ⟷ z3[2.1] ⟷ gnd
end
imp, omega = determine_impedance(net, elim_elements = [:vs],
                            input_pins = Any[:Node1], output_pins= Any[:gnd])
bode(imp, omega = omega)
```

Since the impedance "visible" from the DC source is equal to $1.5s$, the equivalent impedance's magnitude and phase obtained through the simulation are depicted in Fig. 21.

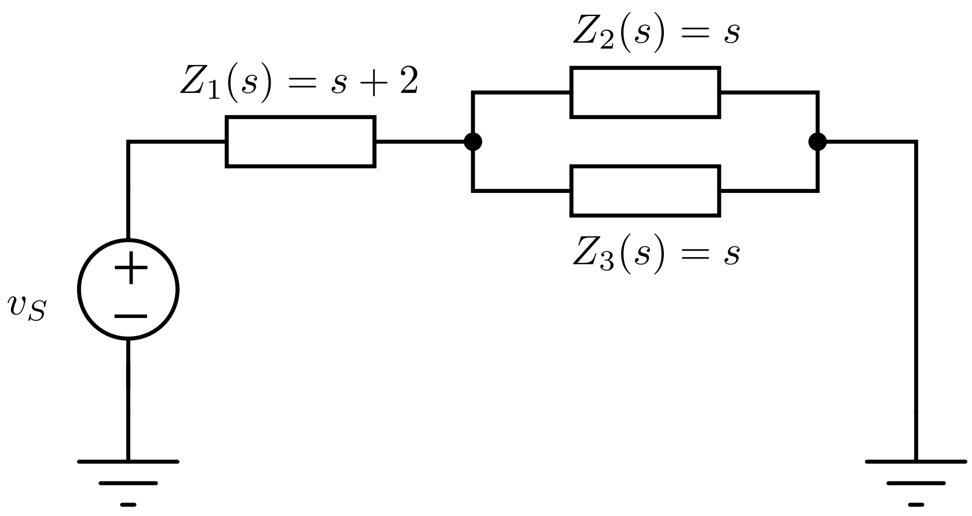


Figure 20: Example of the network with impedances.

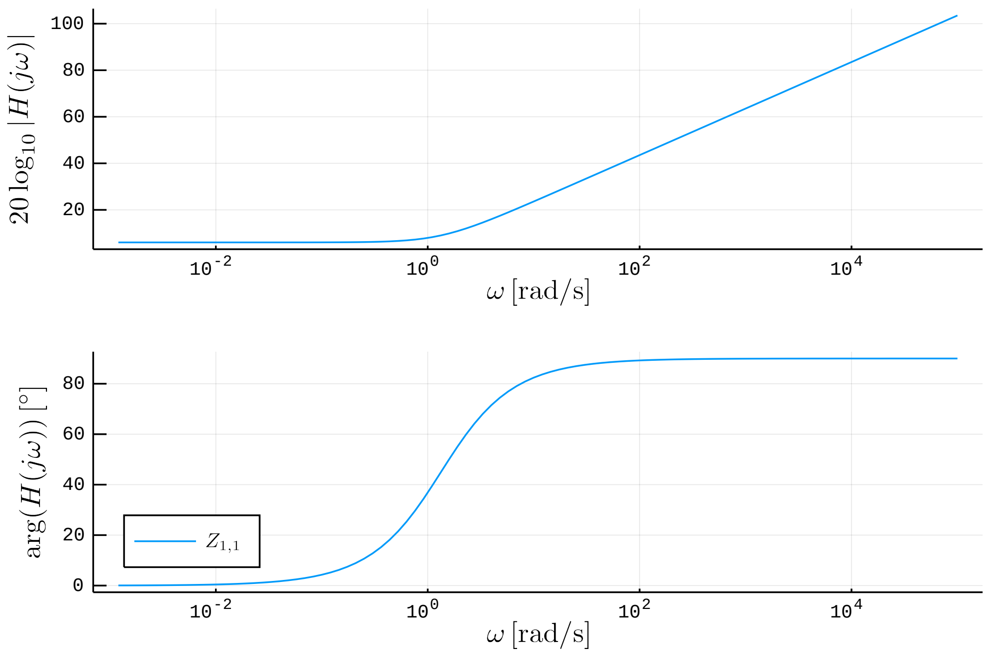


Figure 21: Magnitude and a phase of the equivalent impedance.

4.2.2 Transformer

A transformer element is created using the function `transformer(;args...)`, which creates a dc transformer, a single phase transformer or a three-phase transformer in `YY` or Δ`Y` configuration.

The component is defined using the following structure.

```
@with_kw mutable struct Transformer
    value :: Array{Basic} = []           # ABCD value

    pins :: Int = 1                      # marks single or three phase
    organization :: Symbol = :YY         # three phase organization (:YY or :ΔY)

    ω :: Union{Int, Float64} = 2*π*50    # rated frequency in [Hz]
    V₁ᵒ :: Union{Int, Float64} = 0      # open circuit primary voltage [V]
    V₁ˢ :: Union{Int, Float64} = 0      # short circuit primary voltage [V]
    $I_1^o$ :: Union{Int, Float64} = 0      # open circuit primary current [V]
    $I_1^s$ :: Union{Int, Float64} = 0      # short circuit primary current [V]
    $P_1^o$ :: Union{Int, Float64} = 0      # open circuit losses on primary side [W]
    $P_1^s$ :: Union{Int, Float64} = 0      # short circuit losses on primary side [W]
    $V_2^o$ :: Union{Int, Float64} = 0      # open circuit secondary voltage [V]
    $V_2^s$ :: Union{Int, Float64} = 0      # short circuit secondary voltage [V]

    n :: Union{Int, Float64} = 0          # turn ratio
    $L_p$ :: Union{Int, Float64} = 0       # primary side inductance [H]
    $R_p$ :: Union{Int, Float64} = 0       # primary side resistance [Ω]
    $R_s$ :: Union{Int, Float64} = 0       # secondary side resistance [Ω]
    $L_s$ :: Union{Int, Float64} = 0       # secondary side inductance [H]
    $L_m$ :: Union{Int, Float64} = 0      # magnetising inductance [H]
    $R_m$ :: Union{Int, Float64} = 0      # magnetising resistance [Ω]
    $C_t$ :: Union{Int, Float64} = 0       # turn-to-turn capacitance [F]
    $C_s$ :: Union{Int, Float64} = 0       # stray capacitance [F]
end
```

Pins: 1.1, 2.1 for single phase transformer and 1.1, 1.2, 1.3, 2.1, 2.2, 2.3 for a three-phase transformer

Example: For the circuit containing a single phase transformer and an inductive load, the circuit definition is given in the following listing.

```
s = symbols("s")
net = @network begin
    vs = dc_source(voltage = 5e3)
    t = transformer($V_1^o$ = 2.4e3, $V_1^s$ = 51.87, $V_2^o$ = 240, $P_1^o$ = 171.1, $P_1^s$ = 642.1,
                    $I_1^o$ = 0.48, $I_1^s$ = 20.83, $C_s$ = 12e-6, $C_t$ = 7e-6)
    z = impedance(pins = 1, z = 25e-3*s)

    vs[1.1] ⟷ t[1.1] ⟷ Node1
    t[2.1] ⟷ z[1.1]
    vs[2.1] ⟷ z[2.1] ⟷ gnd
end
```

Impedance “visible” from the voltage source is depicted in Fig. 22.

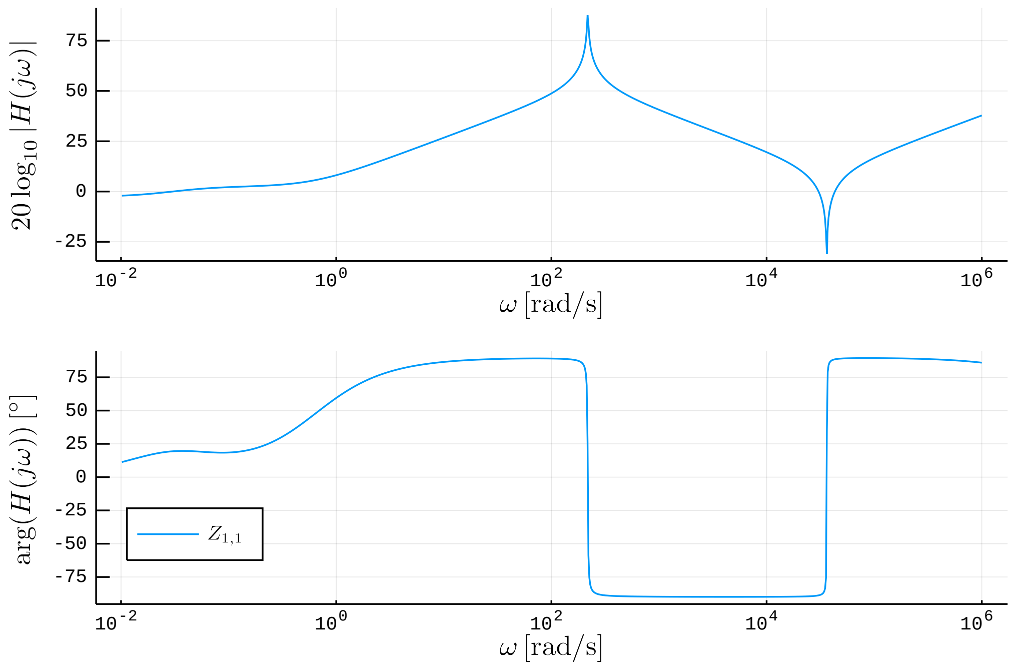


Figure 22: Magnitude and a phase of the equivalent impedance of the network containing a transformer.

4.2.3 Overhead line

Overhead line is defined with a struct and added as a field `element_value` inside the Element struct. It is called with the function: `overhead_line(;args...)`.

Function `overhead_line` generates the element `elem` with the `element_value` of the type `Transmission_line`. Arguments should be given in the form of struct `Overhead_line` fields:

1. length - line length [m]
2. conductors - defined in the following struct

```
struct Conductors
    # number of bundles (phases)
    n^b :: Int = 1
    # number of subconductors per bundle
    n^sb :: Int = 1
    # height above the ground of the lowest bundle  [m]
    y^bc :: Union{Int, Float64}  = 0
    # vertical offset between the bundles [m]
    Δy^bc :: Union{Int, Float64} = 0
    # horizontal offset between the lowest bundles  [m]
    Δx^bc :: Union{Int, Float64} = 0
    # horizontal offset in group of bundles    [m]
    Δx̃^bc :: Union{Int, Float64} = 0
    # sag offset    [m]
    d^sag :: Union{Int, Float64} = 0
    # subconductor spacing (symmetric)  [m]
    d^sb :: Union{Int, Float64} = 0
    # conductor radius  [m]
    r^c :: Union{Int, Float64} = 0
    # DC resistance for the entire conductor [Ω/m]
    R^dc :: Union{Int, Float64} = 0
    # shunt conductance
    g^c :: Union{Int, Float64} = 1e-11
    # relative conductor permeability
    μ_r^c :: Union{Int, Float64} = 1
    # add absolute positions manually
    positions :: Tuple{Vector{Union{Int, Float64}},
                    Vector{Union{Int, Float64}}} = ([],[])
    # can be :flat, :vertical, :delta, :concentric, :offset
    organization :: Symbol = Symbol()
end
```

3. groundwires defined in the following struct

```
struct Groundwires
    # number of groundwires (typically 0 or 2)
    n^g :: Int = 0
    # horizontal offset between groundwires [m]
    Δx^g :: Union{Int, Float64} = 0
    # vertical offset between the lowest conductor and groundwires  [m]
    Δy^g :: Union{Int, Float64} = 0
    # ground wire radius  [m]
    r^g :: Union{Int, Float64} = 0
    # sag offset [m]
    d^gsag ::  Union{Int, Float64} = 0
    # groundwire DC resistance [Ω/m]
    R^gdc :: Union{Int, Float64} = 0
    # relative groundwire permeability
    μ_r^g :: Union{Int, Float64} = 1
    # add absolute positions manually
    positions :: Tuple{Vector{Union{Int, Float64}},
                  Vector{Union{Int, Float64}}} = ([],[])
    eliminate :: Bool = true
end
```

4. `earth_parameters` - with default value (1,1,1) and meaning ($\mu_r\_earth, \epsilon_r\_earth, \rho\_earth$) in units ([], [], [Ωm])

Example:

```
overhead_line(length = 227e3, earth_parameters = (1,1,100),
    conductors = Conductors(n^b = 2, n^sb = 2, organization = :flat,
    R^dc = 0.06266, r^c = 0.01436, y^bc = 27.5, Δx^bc = 11.8, d^sb = 0.4572, d^sag = 10),
    groundwires = Groundwires(n^g = 2, Δx^g = 6.5, Δy^g = 7.5, R^gdc = 0.9196,
    r^g = 0.0062, d^gsag = 10))
```

The short circuit impedance for the transmission line defined in the previous listing is ploted in the Fig. 23.

4.2.4 Cable

Cable is defined with a struct and added as a field `element_value` inside the Element struct. It is called with the function: `cable(;args...)`.

Generates the element `elem` with the `element_value` of the type `Cable`. Arguments should be given in the form of struct `Cable` fields:

1. length - length of the cable in [m]

2. `earth_parameters` - with default value (1,1,1) and meaning ($\mu_r\_earth, \epsilon_r\_earth, \rho\_earth$) in units ([], [], [Ωm]), meaning ground (earth) relative premeability, relative permittivity and ground resistivity

3. conductors - dictionary with the key symbol being: C1, C2, C3 or C4, and the value given with the struct `Conductor`. If the sheath consists of metalic screen and sheath, then add screen with a key symbol SC and sheath with C2.

```
struct Conductor
    r_i :: Union{Int, Float64} = 0            # inner radius
    r_o :: Union{Int, Float64} = 0            # outer radius
    ρ  :: Union{Int, Float64} = 0           # conductor resistivity [Ωm]
    μ_r  :: Union{Int, Float64} = 1          # relative permeability

    A :: Union{Int, Float64} = 0              # nominal area
end
```

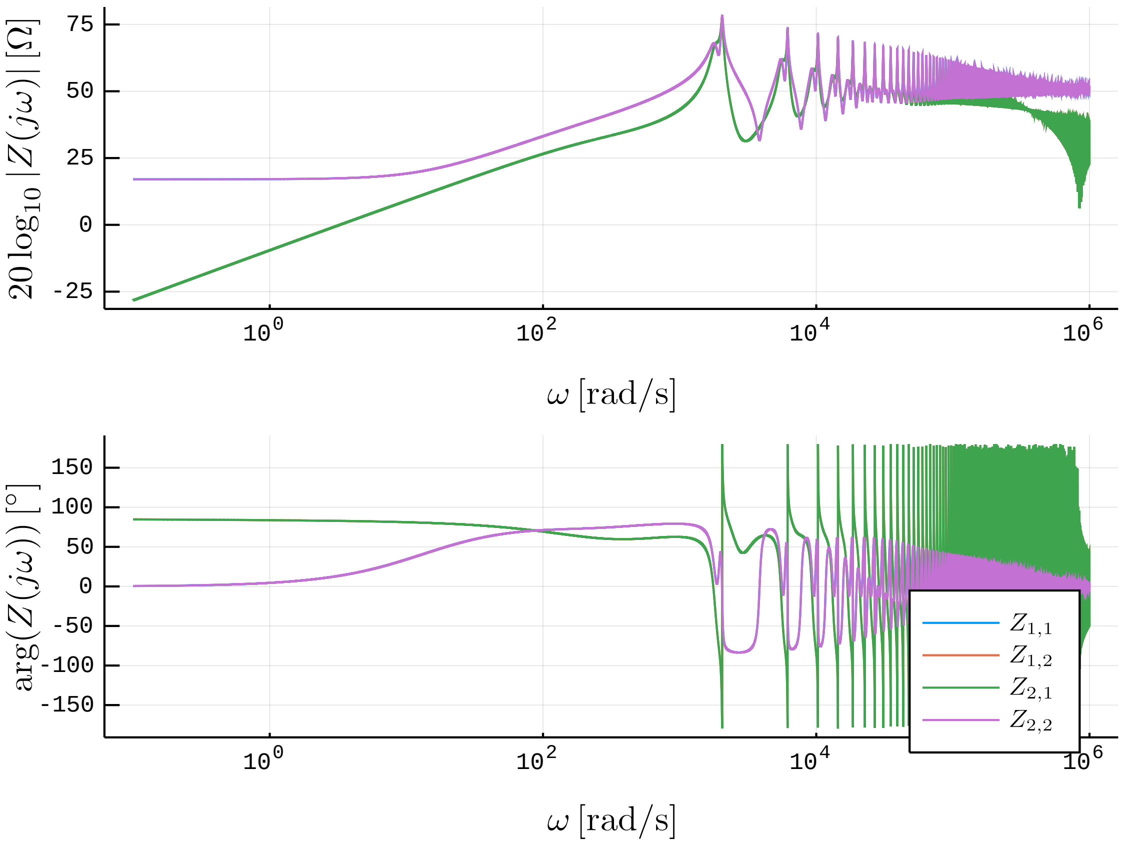


Figure 23: Transmission line example.

4. insulators - dictionary with the key being symbol: I1, I2, I3 and I4, and the value given with the struct `Insulator`. For the insulator 2 the semiconducting layers can be added by specifying outer radius of the inner semiconducting layer and inner radius of the outer semiconducting layer.

```
struct Insulator
    rᵢ :: Union{Int, Float64} = 0                # inner radius
    rₒ :: Union{Int, Float64} = 0                # outer radius
    ϵᵣ :: Union{Int, Float64} = 1                # relative permittivity
    μᵣ :: Union{Int, Float64} = 1                # relative permeability

     # inner semiconductor outer radius
    a :: Union{Int, Float64} = 0
    # outer semiconductor inner radius
    b :: Union{Int, Float64} = 0
end
```

5. positions - given as an array in (x,y) format

6. type - symbol representing aerial or underground cable

7. configuration - symbol with two possible values: coaxial (default) and pipe-type

Example:

```
cable(length = 100e3, positions = [(0,1)],
    C1 = Conductor(rₒ = 24.25e-3, ρ = 1.72e-8),
    C2 = Conductor(rᵢ = 41.75e-3, rₒ = 46.25e-3, ρ = 22e-8),
    C3 = Conductor(rᵢ = 49.75e-3, rₒ = 60.55e-3, ρ = 18e-8, μᵣ = 10),
    I1 = Insulator(rᵢ = 24.25e-3, rₒ = 41.75e-3, ϵᵣ = 2.3),
    I2 = Insulator(rᵢ = 46.25e-3, rₒ = 49.75e-3, ϵᵣ = 2.3),
    I3 = Insulator(rᵢ = 60.55e-3, rₒ = 65.75e-3, ϵᵣ = 2.3))
```

In the Fig. 24 is depicted bode plot for a short-connected cable given in the previous listing with the line length 100 km.

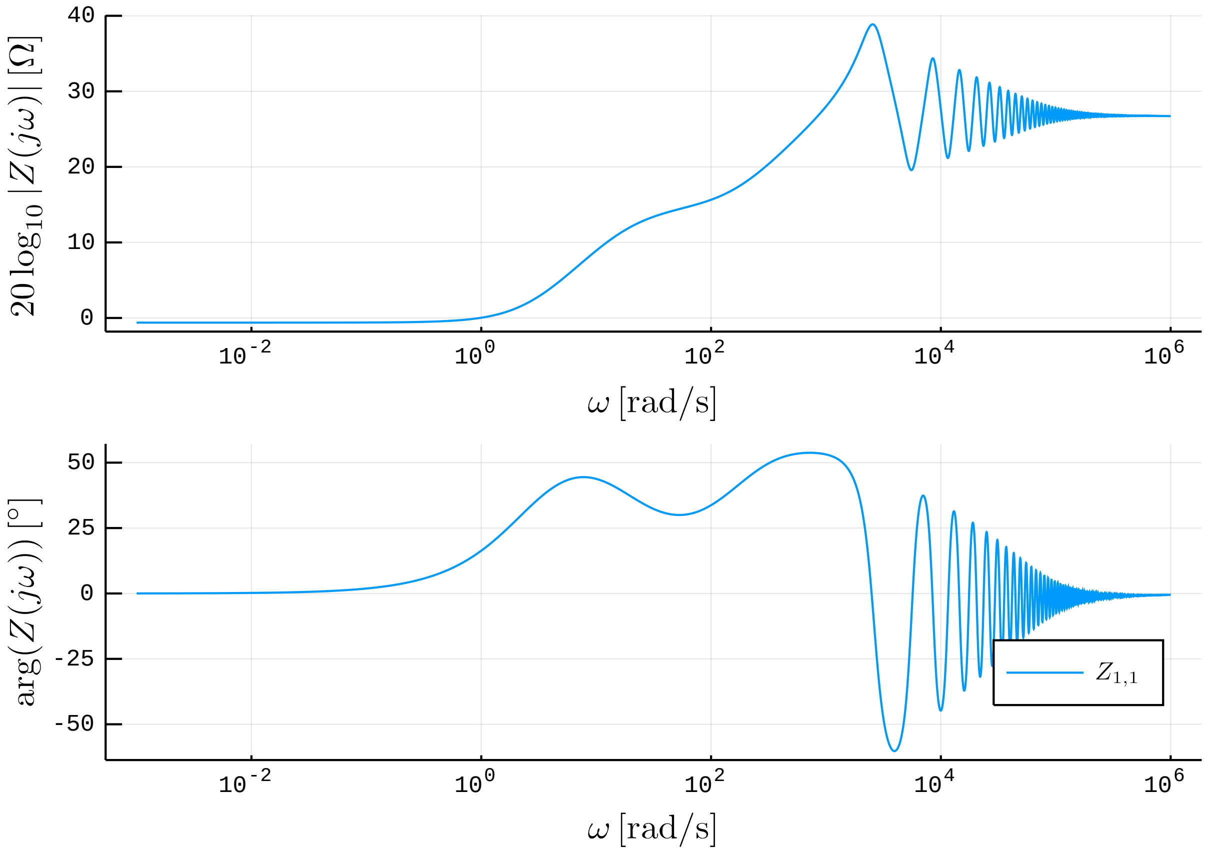


Figure 24: Cable example for line length 100 km.

4.3 MMC

MMC is implemented as `Element` using function `function mmc(;args...)`. Component has then the field `element_value` of a struct type.

```
f₀ :: Union{Int, Float64}  = 50              # nominal frequency
ω₀ :: Union{Int, Float64} = 100*π

S_base :: Union{Int, Float64} = 1000e6       # power rating of the system
V_base :: Union{Int, Float64} = 333e3        # base voltage rating
I_base :: Union{Int, Float64} = 0            # base current rating
Vₘ :: Union{Int, Float64} = 320e3            # AC-grid voltage in abs frame [V]
Vᵈᶜ :: Union{Int, Float64} = 640e3          # DC-bus voltage
V_d^G :: Union{Int, Float64} = 0
V_q^G :: Union{Int, Float64} = 0
```

```
L_arm :: Union{Int, Float64}  = 50e-3        # arm inductance [H]
R_arm :: Union{Int, Float64}  = 1.07         # equivalent arm resistance [Ω]
C_arm :: Union{Int, Float64}  = 10e-3        # capacitance per submodule [F]
N :: Int = 400                               # number of submodules per arm

L_r :: Union{Int, Float64}  = 60e-3          # inductance of the phase reactor [H]
R_r :: Union{Int, Float64}  = 0.535          # resistance of the phase reactor [Ω]

controls :: OrderedDict{Symbol, Controller} = OrderedDict{Symbol, Controller}()
equilibrium :: Array{Union{Int, Float64}} = [0]
A :: Array{Union{Int, Float64}} = [0]
B :: Array{Union{Int, Float64}} = [0]
C :: Array{Union{Int, Float64}} = [0]
D :: Array{Union{Int, Float64}} = [0]
```

The constructed MMC has 2 pins on the AC side: 1.1, 1.2, and 2 pins on its DC-side: 2.1 and 2.2. The component is described using two ABCD parameters with the matrix 4×4.

The controls are defined as `PI_control` and the keyword `occ` is for output current control, `ccc` for circulating current control, `zcc` for zero current control, `power` for active and reactive current control and `energy` for zero energy control.

Example:

For the test example of the MMC with implemented output current control, circulating current control, power and energy controls, using the function:

```
mmc(energy = PI_control(K_p = 120, K_i = 400),
    occ = PI_control(ζ = 0.7, bandwidth = 1000),
    ccc = PI_control(ζ = 0.7, bandwidth = 300),
    zcc = PI_control(ζ = 0.7, bandwidth = 300),
    power = PI_control(K_p = 2.0020e-07, K_i = 1.0010e-04))
```

The generated results obtained for plotting MMC's admittances using function `plot_data` are presented with the Figs. 25, 26 and 27.

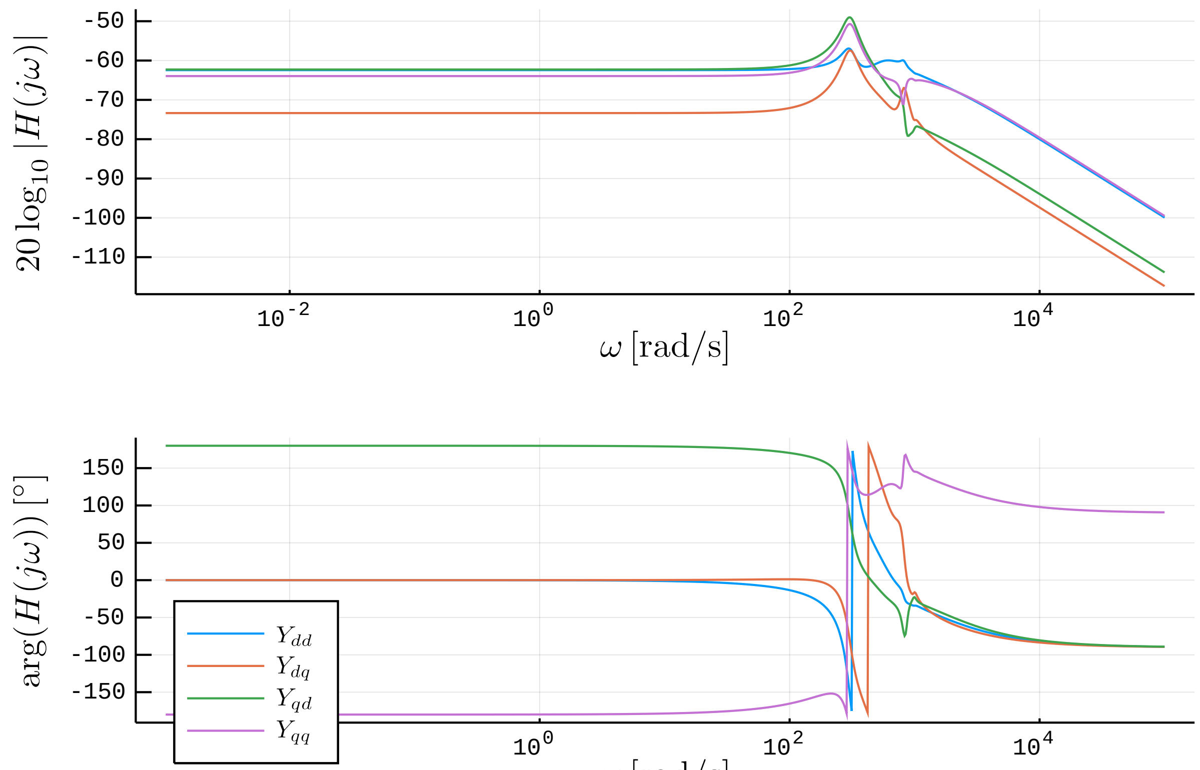


Figure 25: AC side admittance of the MMC.

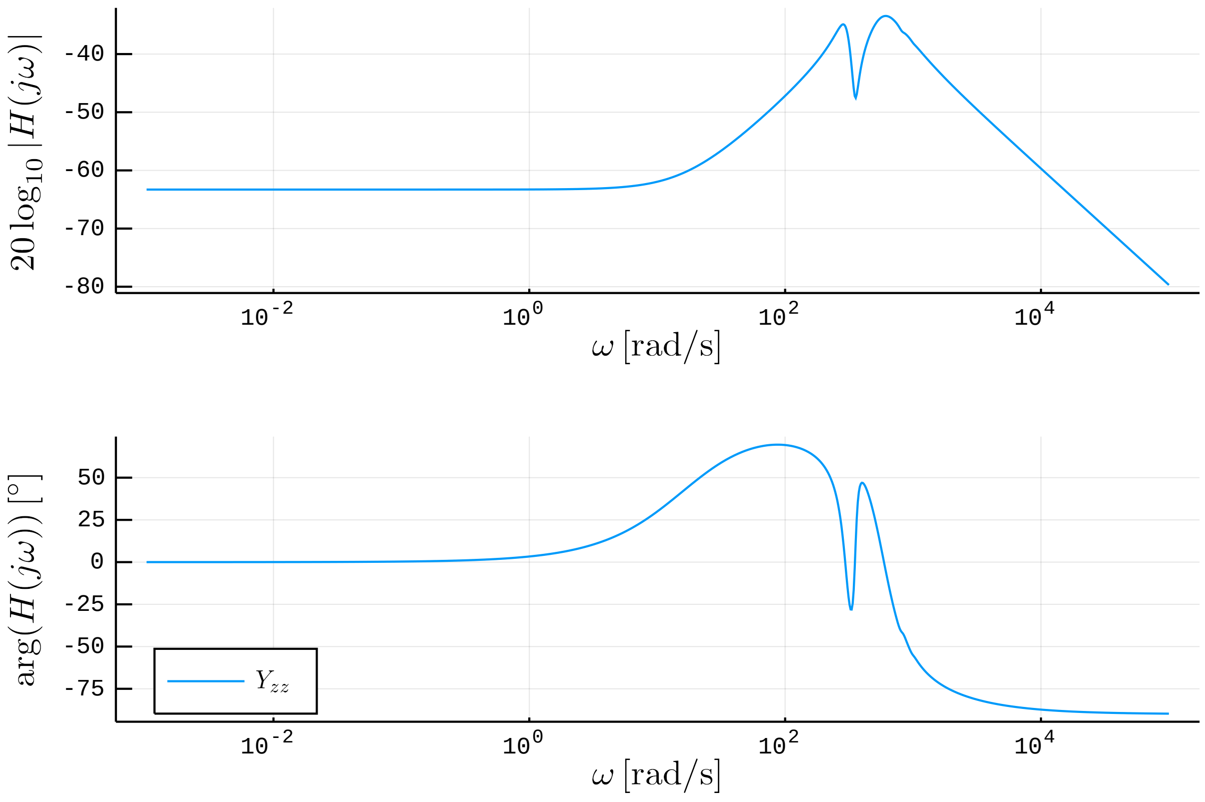


Figure 26: DC side impedance of the MMC.

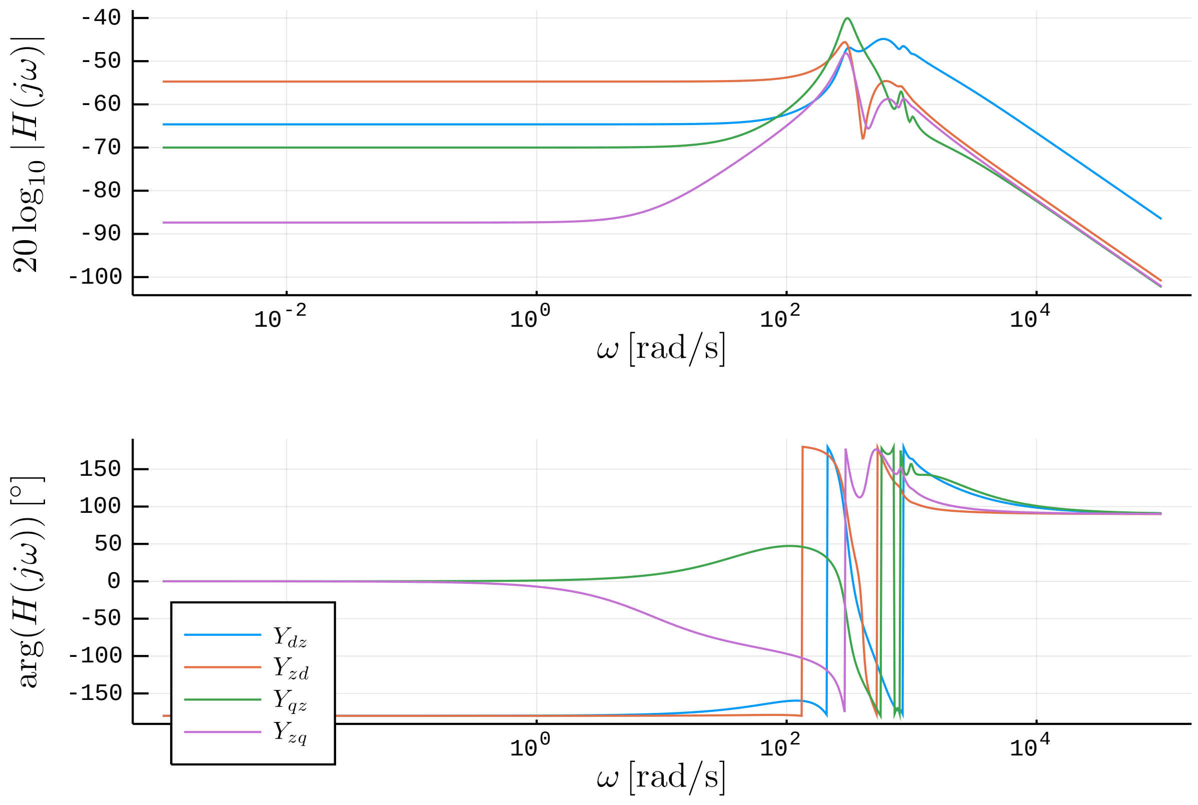


Figure 27: Admittance interconnection between AC and DC side.

For the same test example, but with base values $S_{base} = 1000$ MVA and $V_{base} = 333$ kV, are generated multiple diagrams in order to demonstrate the functionality of the simulator.

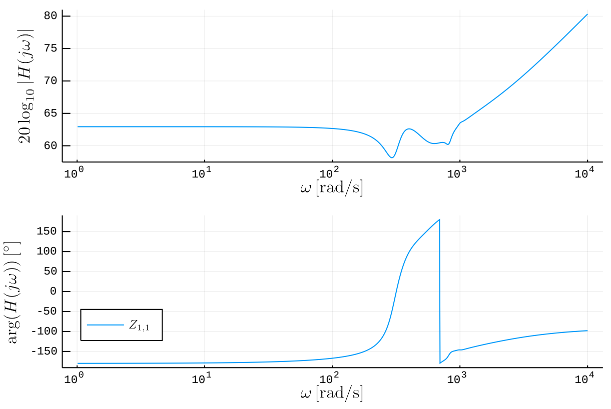
In the case when the impedance between two pins denoted as x and y $(x, y \in \{d, q, z\})$ of three pins is estimated, while the third pin is short connected, the expected impedance is $Z_{eq} = \frac{1}{Y_{xx}}$, where $Y_{xx} \in \{Y_{dd}, Y_{qq}, 3Y_{zz}\}$. The same result is obtained in the simulation and depicted in Fig. 28.

For the case when one pin is considered as "open" and the impedance is determined between other pins, the simulated results are depicted in Fig. 29. The impedance between pins x $(x = d, q)$ and z for the short connected third pin, the expected impedance is:

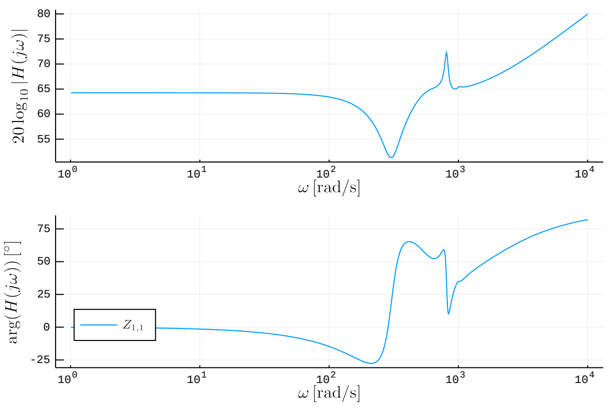
$$Z_{eq} = \frac{1}{3\left(Y_{zz} - \frac{Y_{zx}Y_{xz}}{Y_{xx}}\right)}.$$

Impedance between pins x and y for $x, y = d, q$ with open dc pin z is

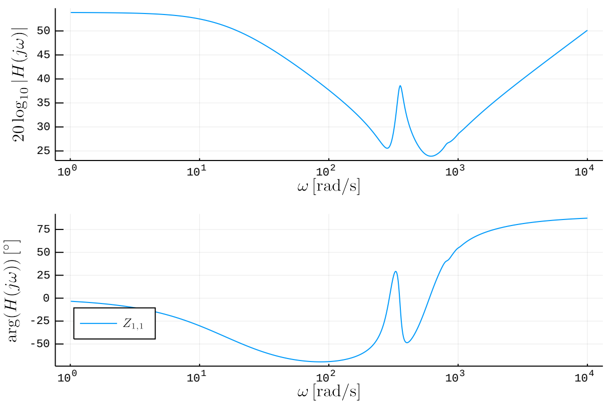
$$Z_{eq} = \frac{1}{Y_{xx} - \frac{Y_{zx}Y_{xz}}{Y_{zz}}}.$$



(a)

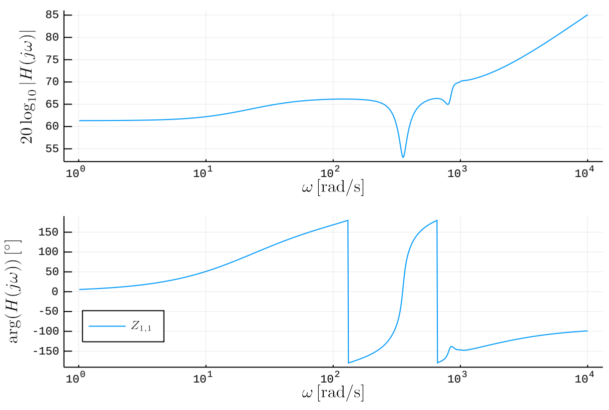


(b)

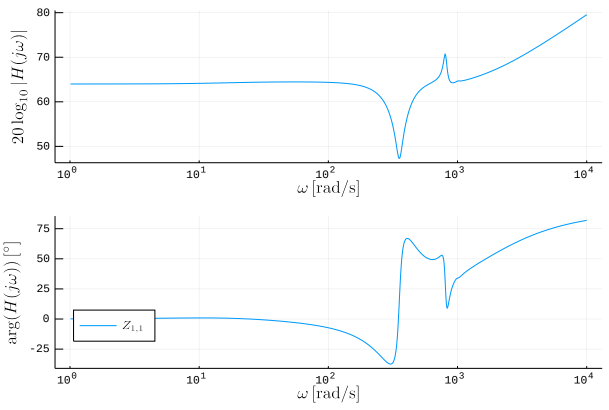


(c)

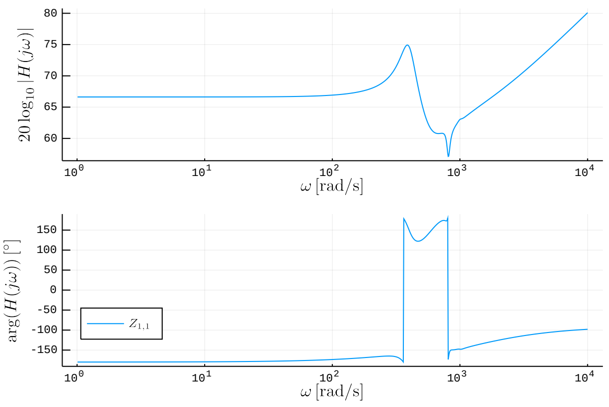
Figure 28: Impedance between two pins when the third is short connected: (a) between d and q pins; (b) between q and d pins; and (c) between dc pin and short connected d and q pins.



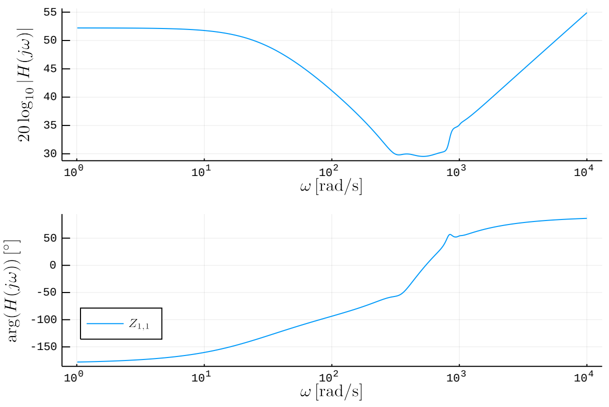
(a)



(b)



(c)



(d)

Figure 29: Impedance between two pins when the third is open connection: (a) between d and q pins; (b) between q and d pins; (c) between z and d; and (d) between z and q.

An example for the simulation of the MMC with an incorporated PLL is given with the following listing.

```
mmc(S_base = 1000e6, V_base = 333e3,
    energy = PI_control(K_p = 120, K_i = 400),
    occ = PI_control(ζ = 0.7, bandwidth = 1000),
    ccc = PI_control(ζ = 0.7, bandwidth = 300),
    zcc = PI_control(ζ = 0.7, bandwidth = 300),
    power = PI_control(K_p = 2.0020e-07, K_i = 1.0010e-04),
    pll = PI_control(K_p = 2e-3, K_i = 2))
```

The generated results obtained for plotting MMC's admittances using function `plot_data` are presented with the Figs. 30, 31 and 32.

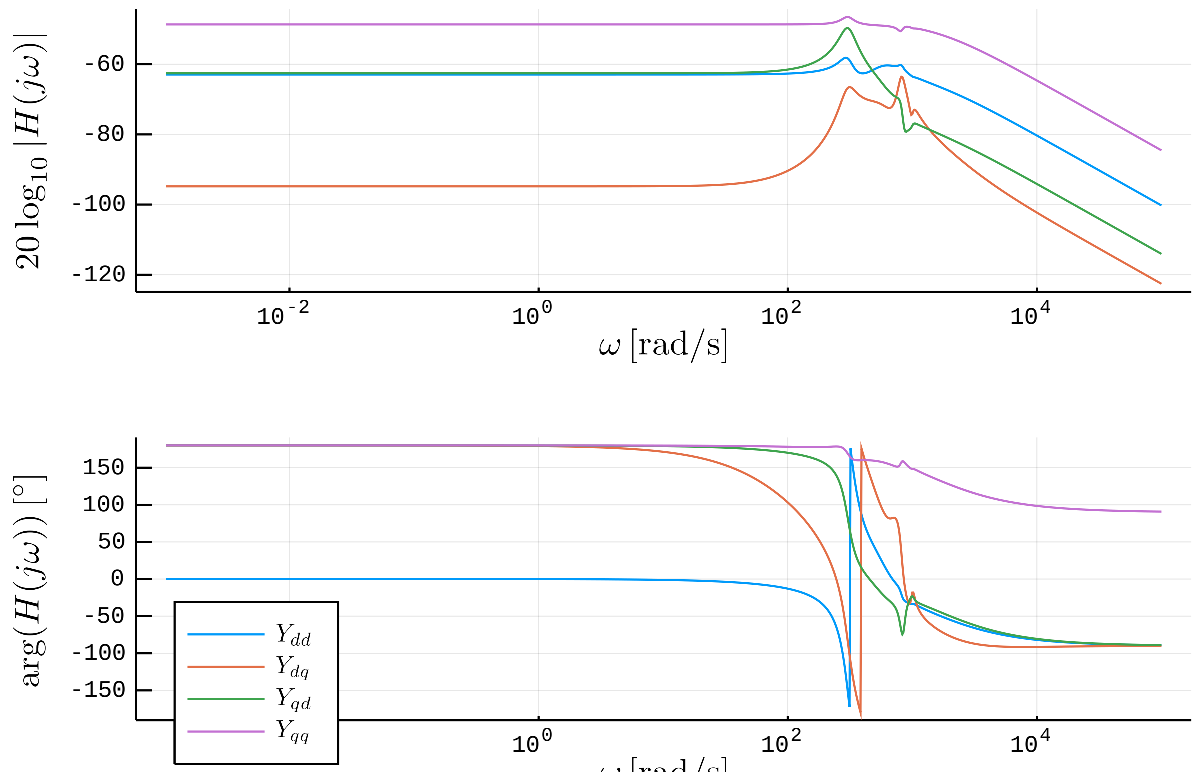


Figure 30: AC side admittance of the MMC with PLL.

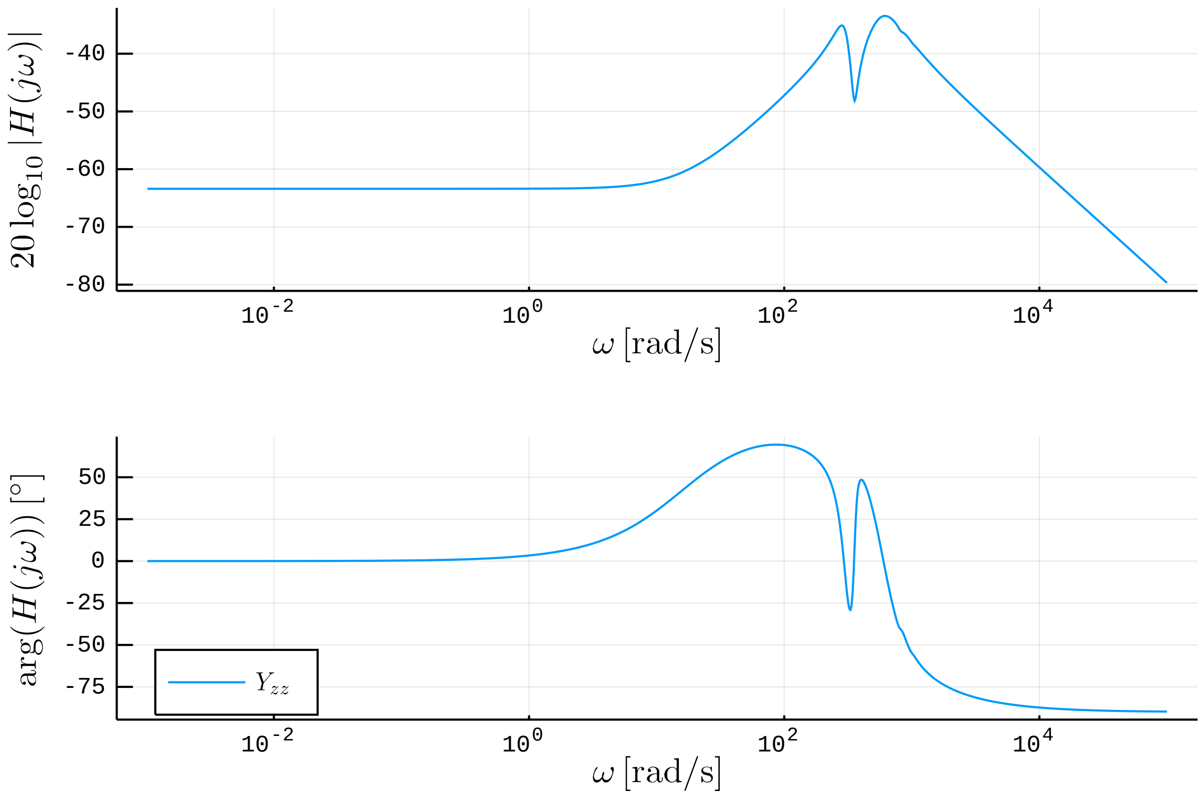


Figure 31: DC side impedance of the MMC with PLL incorporated.

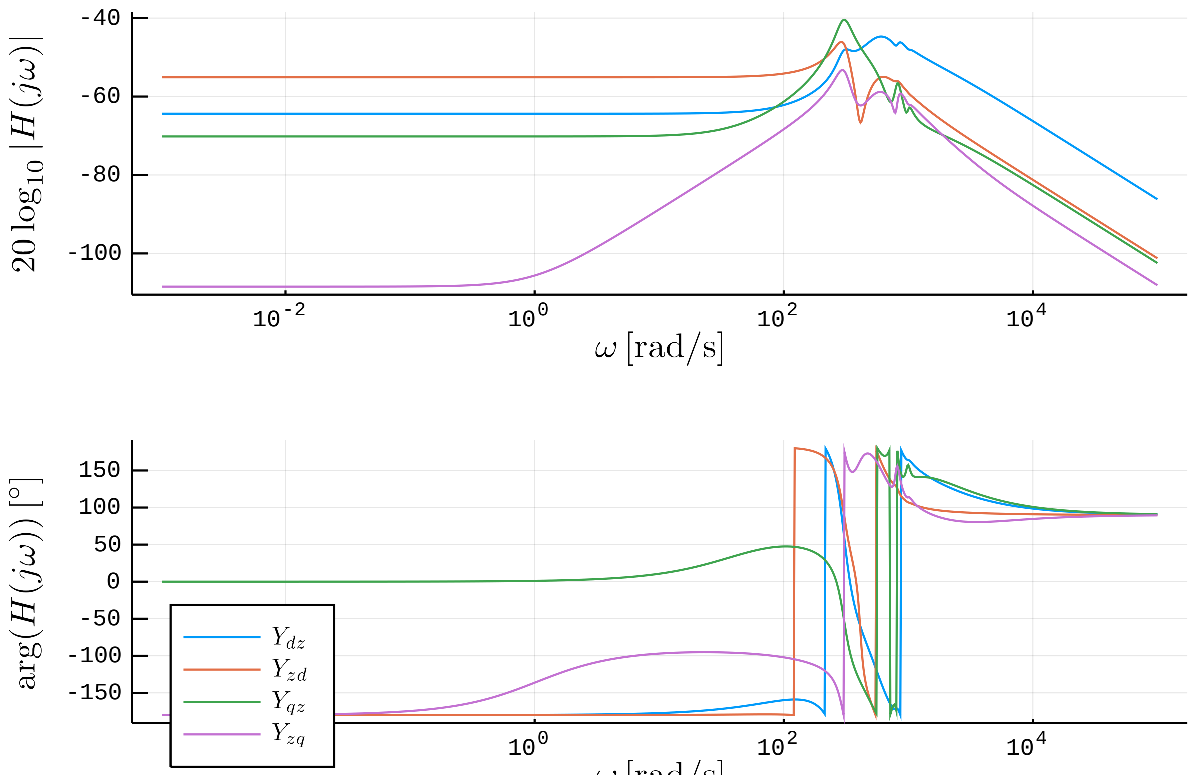


Figure 32: Admittance interconnection between AC and DC side of the MMC with PLL applied.

References

[1] J. A. Martinez-Velasco, *Power system transients: parameter determination.* CRC press, 2017.

[2] F. Castellanos and J. Marti, “Full frequency-dependent phase-domain transmission line model,” *IEEE transactions on Power Systems*, vol. 12, no. 3, pp. 1331–1339, 1997.

[3] A. Morched, B. Gustavsen, and M. Tartibi, “A universal model for accurate calculation of electromagnetic transients on overhead lines and underground cables,” *IEEE Transactions on Power Delivery*, vol. 14, no. 3, pp. 1032–1038, 1999.

[4] H. Manitoba, “Research centre inc,” *PSCAD/EMTDC user’s guide*, 2003.

[5] L. Wu *et al.*, “Impact of ehv/hv underground power cables on resonant grid behavior,” *Eindhoven: Eindhoven University of Technology*, 2014.

[6] A. Ametani, “A general formulation of impedance and admittance of cables,” *IEEE transactions on power apparatus and systems*, no. 3, pp. 902–910, 1980.

[7] P. de Arizon and H. W. Dommel, “Computation of cable impedances based on subdivision of conductors,” *IEEE Transactions on Power Delivery*, vol. 2, no. 1, pp. 21–27, 1987.

[8] F. De Léon, M. L. Márquez-Asensio, and G. Álvarez-Cordero, “Effects of conductor counter-transposition on the positive-sequence impedance and losses of cross-bonded cables,” *IEEE Transactions on Power Delivery*, vol. 26, no. 3, pp. 2060–2063, 2011.

[9] R. A. Rivas and J. R. Marti, “Calculation of frequency-dependent parameters of power cables: Matrix partitioning techniques,” *IEEE transactions on power delivery*, vol. 17, no. 4, pp. 1085–1092, 2002.

[10] G. Bergna-Diaz, J. Freytes, X. Guillaud, S. D’Arco, and J. A. Suul, “Generalized voltage-based state-space modeling of modular multilevel converters with constant equilibrium in steady state,” *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 2, pp. 707–725, 2018.

[11] G. Bergna-Diaz, D. Zonetti, S. Sanchez, R. Ortega, and E. Tedeschi, “Pi passivity-based control and performance analysis of mmc multi-terminal hvdc systems,” *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2018.

[12] Ö. C. Sakinci and J. Beerten, “Generalized dynamic phasor modeling of the mmc for small-signal stability analysis,” *IEEE Transactions on Power Delivery*, vol. 34, no. 3, pp. 991–1000, 2019.

[13] J. Freytes, “Analyse de stabilité en petit signaux des convertisseurs modulaires multiniveaux et application à l’étude d’interopérabilité des mmc dans les réseaux hvdc,” Ph.D. dissertation, Ecole centrale de Lille, 2017.

[14] J. Revels, M. Lubin, and T. Papamarkou, “Forward-mode automatic differentiation in julia,” *arXiv:1607.07892 [cs.MS]*, 2016. [Online]. Available: https://arxiv.org/abs/1607.07892

[15] H. Ergun, J. Dave, D. Van Hertem, and F. Geth, “Optimal power flow for ac/dc grids: Formulation, convex relaxation, linear approximation, and implementation,” *IEEE Transactions on Power Systems*, vol. 34, no. 4, pp. 2980–2990, July 2019.

[16] G. F. Ergun, H. and D. Van Hertem, ““powermodelsacdc repository,” *Department Electrical Engineering, University of Leuven*, 2018. [Online]. Available: https://github.com/hakanergun/PowerModelsACDC.jl

[17] J. Beerten, “Matacdc 1.0 user’s manual,” *Department Electrical Engineering, University of Leuven*, 2012. [Online]. Available: https://www.esat.kuleuven.be/electa/teaching/matacdc/MatACDCManual

[18] R. D. Zimmerman and C. E. Murillo-Sánchez, “Matpower 6.0 user’s manual,” *PSERC: Tempe, AZ, USA*, 2016.