OpFlowTools - simulator design and use

Haixiao Li, Azadeh Kermansaravi, Aleksandra Lekić

March 8, 2025

Contents

1 AC/MTDC Optimal Power Flow

The AC/MTDC optimal power flow (OPF) model presented in this manual is specifically designed for interconnected power systmes consist of multiple AC power grids coupled with a single voltage source converter-based multi-terminal DC (VSC-MTDC) system, as illustrated in Fig. 1.1.

This section provides a concise guide on running codes, modifying input data, and accessing results. It serves as a basic introduction to help to get started with the code.

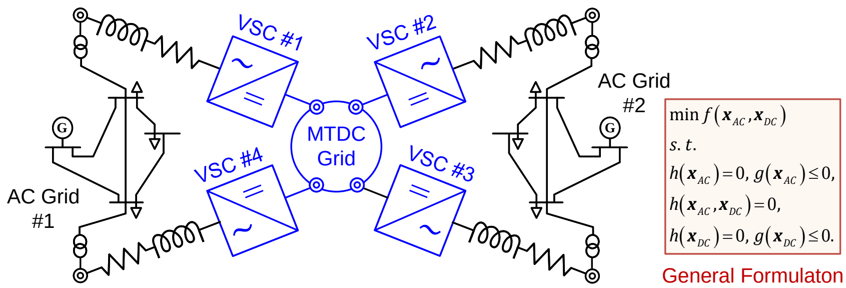

Figure 1.1: Overview of the AC/MTDC OPF problem

1.1 Pacakge Reuqirements

The AC/MTDC optimal power flow (OPF) demo codes are developed using languages: Matlab, Julia, Python, Cpp. For each programming environment, we provide recommendations on the package versions used in our implementation to ensure consistency and reproducibility. While users may install later versions of these packages, it is advised to verify compatibility. Gurobi is used as a unified OPF solver for all language environments. As below, we outline the key dependencies.

- Matlab r2021b: Yalmip 20230622, Matpower 7.1
- Julia v1.10 JuMP.jl 1.23 Gurobi.jl 1.0.0 CSV.jl 0.10.15 DataFrames.jl 1.7.0
- Python v3.12 : Pyomo.py 6.8.2 Gurobipy.py 11.0 Pandas.py 2.2.3 Numpy.py 2.2.1
- Cpp : Gurobi C++ API Eigen 3.4.0
- Gurobi : Gurobi 9.5.2

1.2 Preparing Data Files

The data required for AC/MTDC OPF includes the fundamental data of the ac power grid and the fundamental data of the dc power grid, which are listed and shown in Fig. 1.3. All data are stored in .CSV files due to its compatibility with various language environments.

Users can manually fill in power grid parameters into respective .CSV files based on their specific needs. Below we present the details regarding the respective. CSV. The style of the saved data in .CSV is basically consistent with `mpc` structure used in Matpower.

For the DC grid, the required fundamental data files includes `baseMW_dc.csv`, `pol_dc.csv`, `bus_dc.csv`, `branch_dc.csv`, `branch_dc.csv`. In the file, each column of the data corresponds to a parameter item. A specific rule needs to be followed when saving data into these files. The details about parameter items and descriptions of DC grid data files can be found in Table 1, 2, 3, 4, 5. Note that in `bus_dc.csv` and `branch_dc.csv` files, some parameter items may appear to be meaningless but serve as placeholders to maintain the same format style with the well-known Matpower.

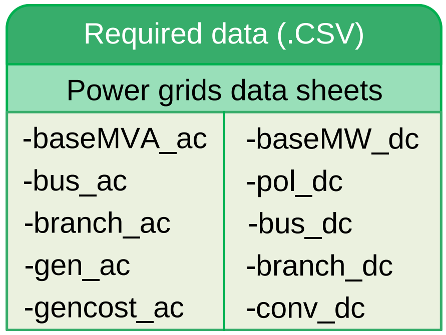

Figure 1.2: Requird power grid data sheet

For the AC grid, the required fundamental data files includes `baseMVA_ac.csv`, `bus_ac.csv`, `branch_ac.csv`, `gen_ac.csv`, `gencost_ac.csv`. The details about parameter items and descriptions of AC grid data files can be found in Table 6, 7, 8, 9, 10. Matpower provides various AC grid case files. Sometimes, users may want to directly select and integrate multiple AC grids without manually modifying data files. To facilitate this process, we provide auxiliary functions `merge_ac.m` and `save_csv.m`. User can build up Matlab script as below.

```
mpc_merged = merge_ac('case9','case14'); % merge_ac('case_a','case_b','...')
save_csv(mpc_merged, directory); % If no saving path is provided, saved in the current directory.
```

After running the script, users can name the prefix of the series of .CSV filenames related to ac grids and directly type it. Finally, save information is shown in command window, like blow.

```
Enter a prefix of the CSV filenames: ac9ac14
Merged AC grid data has already been saved to:
  D:\acdcopf\Tests\ac9ac14_bus.csv
  D:\acdcopf\Tests\ac9ac14_gen.csv
  D:\acdcopf\Tests\ac9ac14_branch.csv
  D:\acdcopf\Tests\ac9ac14_gencost.csv
```

Column	Parameters	Description [Unit]
1	baseMW	Base power capacity [MW].

Table 1: baseMW_dc.csv file

Column	Parameters	Description [Unit]
1	pol	Number of poles (1=monopolar grid, 2=bipolar grid) [-].

Table 2: pol_dc.csv file

Column	Parameters	Description [Unit]
1	bus_i	Bus number [-].
2	—	N/A.
3	Pd	Active power loads [MW].
4	—	N/A.
5	—	N/A.
6	—	N/A.
7	—	N/A.
8	—	N/A.
9	—	N/A.
10	baseKV	Bus voltage level [kV].
11	—	N/A.
12	Vmax	Maximum bus voltage magnitude [p.u.].
13	Vmin	Minimum bus voltage magnitude [p.u.].

Table 3: bus_dc.csv file

Column	Parameters	Description [Unit]
1	fbus	From bus index [-].
2	tbus	To bus index [-].
3	r	Branch resistance [p.u.].
4	—	N/A.
5	—	N/A.
6	—	N/A.
7	—	N/A.
8	—	N/A.
9	—	N/A.
10	—	N/A.
11	—	N/A.
12	—	N/A.
13	—	N/A.

Table 4: branch_dc.csv file

Column	Parameters	Description [Unit]
1	busdc_i	Converter-connected DC bus index [-].
2	busac_i	Converter-connected AC bus index [-].
3	gridac	AC grid number of which converter belongs [-].
4	type_dc	Converter DC-side control modes (1=Constant DC power control, 2=Constant DC voltage control, 3=DC droop control) [-].
5	type_ac	Converter AC-side control modes (1=Constant AC voltage control, 2=Constant reactive power control) [-].
6	P_g	Setting of the active power output of converter DC terminal (applicable to the constant DC power control) [MW].
7	Q_g	Setting of the reactive power output of converter PCC bus (applicable to the constant reactive power control [MVAr].
8	Vtar	Setting of the voltage amplitude of converter DC terminal (applicable to the constant DC voltage control) [p.u.].
9	rtf	Resistance of the converter transformer [p.u.].
10	xtf	Reactance of the converter transformer [p.u.].
11	bf	Conductance of the converter filter [p.u.].
12	rc	Resistance of the converter Phase reactor [p.u.].
13	xc	reactance of the converter Phase reactor [p.u.].
14	basekVac	Base voltage of the converter AC side [p.u.].
15	Vmmax	Maximum voltage magnitude of the converter AC side [p.u.].
16	Vmmin	Minimum voltage magnitude of the converter AC side [p.u.].
17	Imax	Maximum current magnitude of the converter [p.u.].
18	status	Converter status (1=in service, 0=out of service) [p.u.].
19	LossA	Constant loss coefficient [MW].
20	LossB	Linear loss coefficient [kV].
21	LossCR	Rectifier quadratic loss coefficient [Ω].
22	LossCI	Inverter quadratic loss coefficient [Ω].

Table 5: conv_dc.csv file

Column	Parameters	Description [Unit]
1	baseMVA	Base power capacity [MVA].

Table 6: baseMVA_ac.csv file

Column	Parameters	Description [Unit]
1	bus_i	Bus number [-].
2	type	Bus type (3=Slack, 2=PV, 1=PQ) [-]
3	Pd	Active power loads [MW].
4	Qd	Reactive power loads [MVAr].
5	Gs	Shunt conductance (at V=1.0 p.u.) [MW].
6	Bs	Shunt susceptance (at V=1.0 p.u.) [MVAr].
7	area	Area number [-].
8	Vm	Reactive power loads [p.u.].
9	Va	Reactive power loads [drgrees].
10	baseKV	Bus voltage level [kV].
11	zone	Loss zone [-].
12	Vmax	Maximum bus voltage magnitude [p.u.].
13	Vmin	Minimum bus voltage magnitude [p.u.].

Table 7: bus_ac.csv file

Column	Parameters	Description [Unit]
1	fbus	From bus index [-].
2	tbus	To bus index [-].
3	r	Branch resistance [p.u.].
4	x	Branch reactance [p.u.].
5	b	Line charging susceptance.
6	rateA	Rate A (MVA limit for normal operation) [MVA].
7	rateB	Rate B (MVA limit for short-term) [MVA].
8	rateC	Rate C (MVA limit for emergency).
9	ratio	Transformer tap ratio (if applicable).
10	angle	Transformer phase shift angle [degrees].
11	status	Line status (1=active, 0=out-of-service).
12	angmin	Minimum voltage angle difference [-].
13	angmax	Maximum voltage angle difference [-].

Table 8: branch_ac.csv file

Column	Parameters	Description [Unit]
1	bus	Generator's bus number [-].
2	Pg	Active power output [MW].
3	Qg	Reactive power output [MVAr].
4	Qmax	Maximum reactive power output [MVAr].
5	Qmin	Minimum reactive power output [MVAr].
6	Vg	Voltage magnitude setpoint [p.u.].
7	mBase	Total MVA base of generator [MVA].
8	status	Generator status (1=in service, 0=out of service) [-].
9	Pmax	Maximum active power output [MW].
10	Pmin	Minimum active power output [MW].
11	PC1	lower real power output of PQ capability curve [MW].
12	PC2	upper real power output of PQ capability curve [MW].
13	Qc1min	Minimum reactive power output at Pc1 [MVAr].
14	Qc1max	Maximum reactive power at PC1 [MVAr].
15	Qc2min	Minimum reactive power at PC2 [MVAr].
16	Qc2max	Maximum reactive power at PC2 [MVAr].
17	ramp_agc	Ramp rate for automatic generation control [MW/min].
18	ramp_10	Ramp rate for 10-minute reserves [MW].
19	ramp_30	Ramp rate for 30-minute reserves [MW].
20	ramp_q	Ramp rate for reactive power [MVAr/min].
21	apf	Area participation factor [-].

Table 9: gen_ac.csv file

Column	Parameters	Description [Unit]
1	model	Cost function type(1=piecewise, 2=polynomial) [-].
2	startup	Startup cost [$].
3	shuntdown	Shutdown cost [$].
4	n	Number of data points (if piecewise linear) or order of polynomial [$].
5	cost	Coefficients of polynomial (from hgithest to lowest order) or piecewise data points.

Table 10: gencost_ac.csv file

1.3 Running a Simulation

The AC/MTDC OPF program in each language environment have core functions: $\boxed{\texttt{create\_ac}}$-AC system data construction, $\boxed{\texttt{create\_dc}}$-MTDC system data construction, $\boxed{\texttt{params\_ac}}$-AC system parameter extraction, $\boxed{\texttt{params\_dc}}$-DC system parameter extraction, and $\boxed{\texttt{main\_acdcopf}}$-OPF model execution. Users should place all core functions in the same folder and then execute $\boxed{\texttt{main\_acdcopf}}$ to run AC/MTDC OPF. When running $\boxed{\texttt{main\_acdcopf}}$, users need to provide the **prefix names** of the AC and MTDC system .csv files.

1.4 Problem Modeling

This section introduces the modeling of the AC/MTDC power system. The necessary constraint and objective functions to describe the optimal steady-state behavior for the AC/MTDC power system are introduced. Symbols appearing in subsequent formulas can be found below.

1.4.1 Nomenclature

Indexes and Sets

$\bar{(\cdot)}/\underline{(\cdot)}$ -The upper/lower bound of the corresponding optimization variables and parameters

$|\cdot|$ -Magnitude

$\mathcal{L}$ -Branch set

$\mathcal{N}$ -Node set

AC -Alternating current

$cost$ -Generation cost

$droop$ -Droop parameters

gen -Generator

$h,\ i,\ j$ -System node

ij -System branch

$load$ -Load

$loss$ -Power loss

$MTDC$ -Multi-terminal direct current

ref -Reference value

VSC -Voltage source converter

Variables and Parameters Related to the AC Grid

θ_{ij}^{AC} -Phase difference for AC branch ij

$c_{i,2}^{AC}/c_{i,1}^{AC}/c_{i,0}^{AC}$ -Quadratic/Linear/Constant generation coefficient at AC bus i

c_{ij}^{AC}/s_{ij}^{AC} -Optimization variables linked with the AC nodal voltage

p_{ij}^{AC}/q_{ij}^{AC} -Real/Reactive power for AC branch ij

$p_{i,A2V}^{AC}/q_{i,A2V}^{AC}$ -Real/Reactive power delivered from the AC grid to the VSC station at AC bus i

$p_{i,gen}^{AC}/q_{i,gen}^{AC}$ -Real/Reactive power outputs of generator at AC bus i

$p_{i,load}^{AC}/q_{i,load}^{AC}$ -Real/Reactive power loads at AC bus i

p_{i}^{AC}/q_{i}^{AC} -Real/Reactive power injection at AC bus i

v_{i}^{AC} -AC bus i voltage

Variables and Parameters Related to the MTDC Grid

$k_{i,droop}^{MTDC}$ -Droop slope at MTDC bus i

l_{jh}^{MTDC} -Squared current of MTDC branch jh

$p_{i,ref}^{MTDC}$ -Power reference of DC power control/DC droop control at MTDC bus i

p_{jh}^{MTDC} -Power of MTDC branch jh

u_{j}^{MTDC} -Squared MTDC bus j voltage

v_{j}^{MTDC} -MTDC bus j voltage

$V_{i,ref}^{MTDC}$ -Voltage reference of DC voltage control/DC droop control at MTDC bus i

Variables and Parameters Related to the VSC Station

$\theta_{s}^{VSC}/\theta_{f}^{VSC}/\theta_{c}^{VSC}$ -VSC bus $s/f/c$ phase

$a_{loss}^{VSC}/b_{loss}^{VSC}/c_{loss}^{VSC}$ -Constant/Linear/Quadratic coefficient of VSC cpower loss function

b_{f}^{VSC} -Susceptance for VSC bus f to ground

$b_{sf}^{VSC}/b_{fc}^{VSC}$ -Susceptance for VSC branch sf/fc

$c_{sf}^{VSC}/s_{sf}^{VSC}/c_{cf}^{VSC}/s_{cf}^{VSC}$ -Optimization variables linked with the VSC nodal voltage

$g_{sf}^{VSC}/g_{fc}^{VSC}$ -Conductance for VSC branch sf/fc

I_{c}^{VSC} -Phase current at VSC bus c

l_{c}^{VSC} -Squared phase current at VSC bus c

p_{loss}^{VSC} -VSC power loss

$q_{s,ref}^{VSC}$ -Power reference of reactive power control at VSC bus s

$v_{s}^{VSC}/v_{f}^{VSC}/v_{c}^{VSC}$ -VSC bus $s/f/c$ voltage

$v_{s,ref}^{VSC}$ -Voltage reference of AC voltage control at VSC bus s

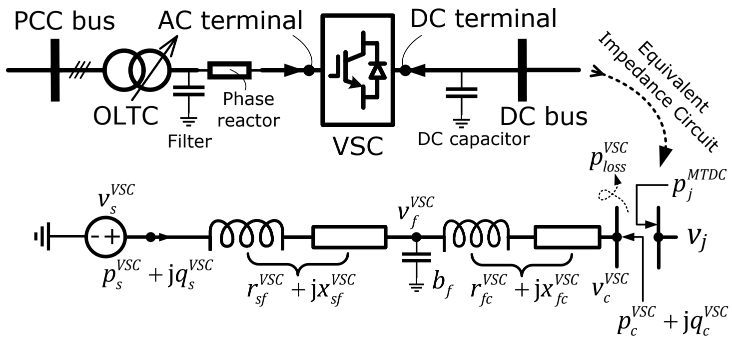

Figure 1.3: Equivalent impedance model of VSC

1.4.2 Original nonlinear formulation

The original AC/MTDC OPF formulation contains is formulated as a nonlinear programming (NLP) problem due to the presence of several nonlinear constraints, such as nonlinear power flow constraints and quadratic converter loss constraints. The explicit formulation of the original AC/MTDC OPF model as below.

Original AC/MTDC OPF Formulation [NLP]

Variables:

$$
\begin{array}{ll}
p_i^{AC}, q_i^{AC}, p_{i,gen}^{AC}, q_{i,gen}^{AC}, p_{i,A2V}^{AC}, q_{i,A2V}^{AC}, \theta_i^{AC}, v_i^{AC} & i \in \mathcal{N}^{AC} \\
p_{ij}^{AC}, q_{ij}^{AC} & (i,j) \in \mathcal{L}^{AC} \\
p_j^{MTDC}, v_j^{MTDC} & j \in \mathcal{N}^{MTDC} \\
p_{jh}^{MTDC} & (j,h) \in \mathcal{L}^{MTDC} \\
p_s^{VSC}, q_s^{VSC}, \theta_s^{VSC}, v_s^{VSC} & s \in \mathcal{N}^{VSC} \\
p_c^{VSC}, q_c^{VSC}, \theta_c^{VSC}, v_c^{VSC}, I_c^{VSC} & c \in \mathcal{N}^{VSC} \\
q_f^{VSC}, \theta_f^{VSC}, v_f^{VSC} & f \in \mathcal{N}^{VSC}
\end{array}
$$

AC constraints:

$$p_{ij}^{AC} = g_{ij}^{AC}\left|v_i^{AC}\right|^2 - \left|v_i^{AC}\right|\left|v_j^{AC}\right|\left(g_{ij}^{AC}\cos\theta_{ij}^{AC} + b_{ij}^{AC}\sin\theta_{ij}^{AC}\right), \quad i,j \in \mathcal{N}^{AC}, \quad (i,j) \in \mathcal{L}^{AC} \tag{2.1a}$$

$$q_{ij}^{AC} = -b_{ij}^{AC}\left|v_i^{AC}\right|^2 + \left|v_i^{AC}\right|\left|v_j^{AC}\right|\left(b_{ij}^{AC}\cos\theta_{ij}^{AC} - g_{ij}^{AC}\sin\theta_{ij}^{AC}\right), \quad i,j \in \mathcal{N}^{AC}, \quad (i,j) \in \mathcal{L}^{AC} \tag{2.1b}$$

$$p_i^{AC} = \sum_{(i,j)} p_{ij}^{AC} + \left(\sum_j G_{ij}^{AC}\right)(v_i^{AC})^2, \quad i,j \in \mathcal{N}^{AC}, \quad (i,j) \in \mathcal{L}^{AC} \tag{2.1c}$$

$$q_i^{AC} = \sum_{(i,j)} q_{ij}^{AC} + \left(\sum_j -B_{ij}^{AC}\right)(v_i^{AC})^2, \quad i,j \in \mathcal{N}^{AC}, \quad (i,j) \in \mathcal{L}^{AC} \tag{2.1d}$$

$$p_i^{AC} = p_{i,gen}^{AC} - p_{i,load}^{AC} - p_{i,A2V}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.2a}$$

$$q_i^{AC} = q_{i,gen}^{AC} - q_{i,load}^{AC} - q_{i,A2V}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.2b}$$

$$\underline{p}_{i,gen}^{AC} \leq p_{i,gen}^{AC} \leq \overline{p}_{i,gen}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.3a}$$

$$\underline{q}_{i,gen}^{AC} \leq q_{i,gen}^{AC} \leq \overline{q}_{i,gen}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.3b}$$

$$\underline{\left|v_i^{AC}\right|} \leq \left|v_i^{AC}\right| \leq \overline{\left|v_i^{AC}\right|}, \quad i \in \mathcal{N}^{AC} \tag{2.4}$$

MTDC constraints:

$$p_{jh}^{MTDC} = \rho^{MTDC} \cdot v_j^{MTDC}\left(v_j^{MTDC} - v_h^{MTDC}\right) y_{jh}^{MTDC}, \quad j,h \in \mathcal{N}^{MTDC}, \quad (j,h) \in \mathcal{L}^{MTDC} \tag{2.5a}$$

$$p_j^{MTDC} = \sum_{(j,h)} p_{jh}^{MTDC}, \quad j,h \in \mathcal{N}^{MTDC}, \quad (j,h) \in \mathcal{L}^{MTDC} \tag{2.5b}$$

$$\underline{v}_j^{MTDC} \leq v_j^{MTDC} \leq \overline{v}_j^{MTDC}, \quad j \in \mathcal{N}^{MTDC} \tag{2.6}$$

$$p_j^{MTDC} = p_{j,ref}^{MTDC}, \quad j \in \mathcal{N}^{MTDC} \tag{2.7a}$$

$$v_j^{MTDC} = v_{j,ref}^{MTDC}, \quad j \in \mathcal{N}^{MTDC} \tag{2.7b}$$

$$p_j^{MTDC} = p_{j,ref}^{MTDC} - \frac{1}{k_{j,droop}^{MTDC}}\left(v_j^{MTDC} - v_{j,ref}^{MTDC}\right), \quad j \in \mathcal{N}^{MTDC} \tag{2.7c}$$

VSC constraints:

$$p_s^{VSC} = p_{i,A2V}^{AC}, \quad s \in \mathcal{N}^{VSC}, \quad i \in \mathcal{N}^{AC} \tag{2.8a}$$

$$q_s^{VSC} = q_{i,A2V}^{AC}, \quad s \in \mathcal{N}^{VSC}, \quad i \in \mathcal{N}^{AC} \tag{2.8b}$$

$$\begin{aligned} p_s^{VSC} = \left|v_s^{VSC}\right|^2 g_{sf}^{VSC} - \left|v_s^{VSC}\right|\left|v_f^{VSC}\right| \left[g_{sf}^{VSC} \cos\left(\theta_s^{VSC} - \theta_f^{VSC}\right)\right. \\ \left. + b_{sf}^{VSC} \sin\left(\theta_s^{VSC} - \theta_f^{VSC}\right)\right], \quad s, f \in \mathcal{N}^{VSC} \end{aligned} \tag{2.9a}$$

$$\begin{aligned} q_s^{VSC} = -\left|v_s^{VSC}\right|^2 b_{sf}^{VSC} - \left|v_s^{VSC}\right|\left|v_f^{VSC}\right| \left[g_{sf}^{VSC} \cos\left(\theta_s^{VSC} - \theta_f^{VSC}\right)\right. \\ \left. - b_{sf}^{VSC} \sin\left(\theta_s^{VSC} - \theta_f^{VSC}\right)\right], \quad s, f \in \mathcal{N}^{VSC} \end{aligned} \tag{2.9b}$$

$$\begin{aligned} p_c^{VSC} = \left|v_c^{VSC}\right|^2 g_{fc}^{VSC} + \left|v_f^{VSC}\right|\left|v_c^{VSC}\right| \left[g_{fc}^{VSC} \cos\left(\theta_f^{VSC} - \theta_c^{VSC}\right)\right. \\ \left. - b_{fc}^{VSC} \sin\left(\theta_f^{VSC} - \theta_c^{VSC}\right)\right], \quad f, c \in \mathcal{N}^{VSC} \end{aligned} \tag{2.10a}$$

$$\begin{aligned} q_c^{VSC} = -\left|v_c^{VSC}\right|^2 b_{fc}^{VSC} + \left|v_f^{VSC}\right|\left|v_c^{VSC}\right| \left[g_{fc}^{VSC} \sin\left(\theta_f^{VSC} - \theta_c^{VSC}\right)\right. \\ \left. + b_{fc}^{VSC} \cos\left(\theta_f^{VSC} - \theta_c^{VSC}\right)\right], \quad f, c \in \mathcal{N}^{VSC} \end{aligned} \tag{2.10b}$$

$$q_f^{VSC} = -\left|v_f^{VSC}\right|^2 b_f^{VSC}, \quad f \in \mathcal{N}^{VSC} \tag{2.11}$$

$$p_c^{VSC} + p_{loss}^{VSC} + p_j^{MTDC} = 0, \quad j \in \mathcal{N}^{MTDC}, \quad c \in \mathcal{N}^{VSC} \tag{2.12}$$

$$p_{loss}^{VSC} = a_{loss}^{VSC} + b_{loss}^{\text{VSC}} \left|I_c^{VSC}\right| + c_{loss}^{VSC} \left|I_c^{VSC}\right|^2, \quad c \in \mathcal{N}^{VSC} \tag{2.13a}$$

$$0 \leq \left|I_c^{VSC}\right| \leq \left|\overline{I_c^{VSC}}\right|, \quad c \in \mathcal{N}^{VSC} \tag{2.13b}$$

$$(v_c^{VSC})^2 \cdot \left|I_c^{VSC}\right|^2 = (p_c^{VSC})^2 + (q_c^{VSC})^2 \quad c \in \mathcal{N}^{VSC} \tag{2.14}$$

$$v_s^{VSC} = v_{s,ref}^{VSC}, \quad s \in \mathcal{N}^{VSC} \tag{2.15a}$$

$$q_s^{VSC} = q_{s,ref}^{VSC}, \quad s \in \mathcal{N}^{VSC} \tag{2.15b}$$

Goal:

$$\min \sum_i c_{i,2}^{AC} \left(p_{i,gen}^{AC}\right)^2 + c_{i,1}^{AC} p_{i,gen}^{AC} + c_{i,0}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.16}$$

- **AC Constraints**: ✓(2.1a)-(2.1d) are power flow constraints. ✓(2.2a)-(2.2b) are nodal power balance equations. ✓(2.3a)-(2.3b) are generator power output limits. (2.4) is the voltage magnitude limit.
- **MTDC Constraints**: ✓(2.5a)-(2.5b) are power flow constraints. ✓(2.6) is the voltage magnitude limit. ✓(2.7a)-(2.7c) represents the VSC control mode at DC terminal. Note that only one of them is hold depending on the chosen control model.
- **VSC Constraint**: ✓(2.8a)-(2.8b) are the active and reactive power couplings between the AC grid and VSC. ✓(2.9a)-(2.9b) are the active and reactive power injections from the view of node s.✓(2.10a)-(2.10b) are the active and reactive power injections from the view of node c. ✓(2.11) is the reactive power injection from the view of node f. If we consider the latest converters with the MMC topology, the filter is not required to install at node f and $b_f^{vsc} = 0$.✓(2.12) is the active power coupling between the MTDC grid and VSC.✓(2.13a)-(2.13b) are related to the converter loss function. ✓(2.14) denote the voltage and current relationship from the view of node c. ✓(2.15a)-(2.15b) represent the VSC control mode at AC PCC. Note that only one of them holds depending on the chosen control model.
- **Goal**: ✓(2.16) represent minimizing power generation costs.

1.4.3 Convex relaxed formulation

The original AC/DC OPF problem is NLP. Such a problem is not favored due to their limited scalability and computational challenges. Give these reasons, we formulate a convex relaxed version of the original AC/DC OPF problem. Our demo code is based on the convex relaxed version as below.

Convex Relaxed AC/MTDC OPF Formulation [SOCP]

Variables:

$p_i^{AC}, q_i^{AC}, p_{i,gen}^{AC}, q_{i,gen}^{AC}, p_{i,A2V}^{AC}, q_{i,A2V}^{AC}, s_{ij}^{AC}, c_{ij}^{AC}$	$i, j \in \mathcal{N}^{AC}$	
p_{ij}^{AC}, q_{ij}^{AC}	$(i, j) \in \mathcal{L}^{AC}$	
p_j^{MTDC}, v_j^{MTDC}	$j \in \mathcal{N}^{MTDC}$	
$p_{jh}^{MTDC}, l_{jh}^{MTDC}$	$(j, h) \in \mathcal{L}^{MTDC}$	
p_s^{VSC}, q_s^{VSC}	$s \in \mathcal{N}^{MTDC}$	
$p_c^{VSC}, q_c^{VSC}, I_c^{VSC}, l_c^{VSC}$	$c \in \mathcal{N}^{MTDC}$	
q_f^{VSC}	$f \in \mathcal{N}^{MTDC}$	
$c_{mn}^{VSC}, c_{mn}^{VSC}$	$m, n \in \{s, f, c\}$	$s, f, c \in \mathcal{N}^{MTDC}$

AC Constraints:

$$p_{ij}^{AC} = G_{ij}\left(c_{ii}^{AC} - c_{ij}^{AC}\right) + B_{ij}s_{ij}^{AC}, \quad i, j \in \mathcal{N}^{AC}, \quad (i, j) \in \mathcal{L}^{AC} \tag{2.17a}$$

$$q_{ij}^{AC} = -B_{ij}\left(c_{ii}^{AC} - c_{ij}^{AC}\right) + G_{ij}s_{ij}^{AC}, \quad i, j \in \mathcal{N}^{AC}, \quad (i, j) \in \mathcal{L}^{AC} \tag{2.17b}$$

$$p_i^{AC} = c_{ii}^{AC}G_{ii} + \sum_j \left(c_{ij}^{AC}G_{ij} - s_{ij}^{AC}B_{ij}\right), \quad i, j \in \mathcal{N}^{AC} \tag{2.17c}$$

$$q_i^{AC} = -c_{ii}^{AC}B_{ii} - \sum_j \left(c_{ij}^{AC}B_{ij} + s_{ij}^{AC}G_{ij}\right), \quad i, j \in \mathcal{N}^{AC} \tag{2.17d}$$

$$c_{ij}^{AC} = c_{ji}^{AC}, \quad i, j \in \mathcal{N}^{AC} \tag{2.17e}$$

$$s_{ij}^{AC} = -s_{ji}^{AC}, \quad i, j \in \mathcal{N}^{AC} \tag{2.17f}$$

$$(c_{ij}^{AC})^2 + (s_{ij}^{AC})^2 \leq c_{ii}^{AC}c_{jj}^{AC}, \quad i, j \in \mathcal{N}^{AC} \tag{2.17g}$$

$$p_i^{AC} = p_{i,gen}^{AC} - p_{i,load}^{AC} - p_{i,A2V}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.18a}$$

$$q_i^{AC} = q_{i,gen}^{AC} - q_{i,load}^{AC} - q_{i,A2V}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.18b}$$

$$\underline{p}_{i,gen}^{AC} \leq p_{i,gen}^{AC} \leq \overline{p}_{i,gen}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.19a}$$

$$\underline{q}_{i,gen}^{AC} \leq q_{i,gen}^{AC} \leq \overline{q}_{i,gen}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.19b}$$

$$\underline{\left|v_i^{AC}\right|}^2 \leq c_{ii}^{AC} \leq \overline{\left|v_i^{AC}\right|}^2, \quad i \in \mathcal{N}^{AC} \tag{2.20}$$

MTDC constraints:

$$p_j^{MTDC} = \sum_{(j,h)} p_{jh}^{MTDC}, \quad j \in \mathcal{N}^{MTDC}, \quad (j, h) \in \mathcal{L}^{MTDC} \tag{2.21a}$$

$$p_{jh}^{MTDC} + p_{hj}^{MTDC} = r_{jh}^{MTDC}l_{jh}^{MTDC}, \quad (j, h) \in \mathcal{L}^{MTDC} \tag{2.21b}$$

$$u_j^{MTDC} - u_h^{MTDC} = r_{jh}^{MTDC}\left(p_{jh}^{MTDC} - p_{hj}^{MTDC}\right), \quad j, h \in \mathcal{N}^{MTDC}, \quad (j, h) \in \mathcal{L}^{MTDC} \tag{2.21c}$$

$$p_{jh}^{MTDC} \leq l_{jh}^{MTDC}u_j^{MTDC} \quad j \in \mathcal{N}^{MTDC}, \quad (j, h) \in \mathcal{L}^{MTDC} \tag{2.21d}$$

$$\underline{u}_j^{MTDC} \leq u_j^{MTDC} \leq \overline{u}_j^{MTDC}, \quad j \in \mathcal{N}^{MTDC} \tag{2.22}$$

$$p_j^{MTDC} = p_{j,ref}^{MTDC}, \quad j \in \mathcal{N}^{MTDC} \tag{2.23a}$$

$$u_j^{MTDC} = \left(v_{j,ref}^{MTDC}\right)^2, \quad j \in \mathcal{N}^{MTDC} \tag{2.23b}$$

$$p_j^{MTDC} = p_{j,ref}^{MTDC} - \frac{1}{k_{j,droop}^{MTDC}}\left(\frac{1}{2} + \frac{1}{2}u_j^{MTDC} - v_{j,ref}^{MTDC}\right), \quad j \in \mathcal{N}^{MTDC} \tag{2.23c}$$

VSC constraints:

$$p_s^{VSC} = p_{i,A2V}^{AC}, \quad s \in \mathcal{N}^{VSC} \tag{2.24a}$$
$$q_s^{VSC} = q_{i,A2V}^{AC}, \quad s \in \mathcal{N}^{VSC} \tag{2.24b}$$
$$p_s^{VSC} = c_{ss}^{VSC} g_{sf}^{VSC} - c_{sf}^{VSC} g_{sf}^{VSC} + s_{sf}^{VSC} b_{sf}^{VSC}, \quad s, f \in \mathcal{N}^{MTDC} \tag{2.25a}$$
$$q_s^{VSC} = -c_{ss}^{VSC} b_{sf}^{VSC} + c_{sf}^{VSC} b_{sf}^{VSC} + s_{sf}^{VSC} g_{sf}^{VSC}, \quad s, f \in \mathcal{N}^{MTDC} \tag{2.25b}$$
$$c_{sf}^{VSC} = c_{fs}^{VSC}, \quad s, f \in \mathcal{N}^{AC} \tag{2.25c}$$
$$s_{sf}^{VSC} = -s_{fs}^{VSC}, \quad s, f \in \mathcal{N}^{VSC} \tag{2.25d}$$
$$(c_{sf}^{VSC})^2 + (s_{sf}^{VSC})^2 \leq c_{ss}^{VSC} c_{ff}^{VSC}, \quad s, f \in \mathcal{N}^{AC} \tag{2.25e}$$
$$p_c^{VSC} = c_{cc}^{VSC} g_{cf}^{VSC} - c_{cf}^{VSC} g_{cf}^{VSC} + s_{cf}^{VSC} b_{cf}^{VSC}, \quad c, f \in \mathcal{N}^{MTDC} \tag{2.26a}$$
$$q_c^{VSC} = -c_{cc}^{VSC} b_{cf}^{VSC} + c_{cf}^{VSC} b_{cf}^{VSC} + s_{cf}^{VSC} g_{cf}^{VSC}, \quad c, f \in \mathcal{N}^{MTDC} \tag{2.26b}$$
$$c_{cf}^{VSC} = c_{fc}^{VSC}, \quad c, f \in \mathcal{N}^{VSC} \tag{2.26c}$$
$$s_{cf}^{VSC} = -s_{fc}^{VSC}, \quad c, f \in \mathcal{N}^{VSC} \tag{2.26d}$$
$$(c_{cf}^{VSC})^2 + (s_{cf}^{VSC})^2 \leq c_{cc}^{VSC} c_{ff}^{VSC}, \quad c, f \in \mathcal{N}^{VSC} \tag{2.26e}$$
$$q_f^{VSC} = -c_{ff}^{VSC} b_f^{VSC}, \quad f \in \mathcal{N}^{VSC} \tag{2.27}$$
$$p_c^{VSC} + p_{loss}^{VSC} + p_j^{MTDC} = 0, \quad j \in \mathcal{N}^{MTDC}, \quad c \in \mathcal{N}^{VSC} \tag{2.28}$$
$$p_{loss}^{VSC} = a_{loss}^{VSC} + b_{loss}^{\text{V S C}} \left|I_c^{VSC}\right| + c_{loss}^{VSC} l_c^{VSC}, \quad c \in \mathcal{N}^{VSC} \tag{2.29a}$$
$$0 \leq \left|I_c^{VSC}\right| \leq \left|\overline{I_c^{VSC}}\right|, \quad c \in \mathcal{N}^{VSC} \tag{2.29b}$$
$$0 \leq l_c^{VSC} \leq \left|\overline{I_c^{VSC}}\right|^2, \quad c \in \mathcal{N}^{VSC} \tag{2.29c}$$
$$\left|I_c^{VSC}\right|^2 \leq l_c^{VSC}, \quad c \in \mathcal{N}^{VSC} \tag{2.30a}$$
$$(p_c^{VSC})^2 + (q_c^{VSC})^2 \leq c_{cc}^{VSC} \cdot l_c^{VSC} \quad c \in \mathcal{N}^{VSC} \tag{2.30b}$$
$$c_{ss}^{VSC} = (v_{s,ref}^{VSC})^2, \quad s \in \mathcal{N}^{VSC} \tag{2.31a}$$
$$q_s^{VSC} = q_{s,ref}^{VSC}, \quad s \in \mathcal{N}^{VSC} \tag{2.31b}$$

Goal:

$$\min \sum_i c_{i,2}^{AC} \left(p_{i,gen}^{AC}\right)^2 + c_{i,1}^{AC} p_{i,gen}^{AC} + c_{i,0}^{AC}, \quad i \in \mathcal{N}^{AC} \tag{2.32}$$

- **AC Constraints**: ✓(2.17a)-(2.17g) consist of the second-order cone (SOC) relaxed power flow constraints [1]. ✓(2.18a)-(2.18b) are nodal power balance equations. ✓(2.19a)-(2.19b) are generator power output limits. (2.20) is the squared voltage magnitude limit.
- **MTDC Constraints**: ✓(2.21a)-(2.21d) consist of the SOC relaxed power flow constraints [2]. ✓(2.22) is the squared voltage magnitude limit. ✓(2.23a)-(2.23c) represents the VSC control mode at DC terminal. Especially, the term v_{ij}^{MTDC} is replaced by the expression with respect to u_{ij}^{MTDC} in (2.23c), through Talyor expansion. Note that for (2.23a)-(2.23c), only one of them holds depending on the chosen control model.
- **VSC Constraint**: ✓(2.24)-(2.24b) are the active and reactive power couplings between the AC grid and VSC. ✓(2.25a)-(2.25e) consist of the SOC relaxed active and reactive power injections from the view of node s.✓(2.26a)-(2.26e) consist of the SOC relaxed active and reactive power injections from the view of node c. ✓(2.27) is the reactive power injection from the view of node f. ✓(2.28) is the active power coupling between the MTDC grid and VSC. ✓(2.29a)-(2.29c) are related to the converter loss function. ✓(2.30a)-(2.30b) consist of the SOC relaxed voltage and current relationship from the view of node c. ✓(2.31a)-(2.31b) represent the VSC control mode at AC PCC. Note that only one of them is hold depending on the chosen control model.

- **Goal**: ✓(2.32) represent minimizing power generation costs.

1.5 Cross-Language Implementation

To help users better understand the above OPF modeling in different language environments, we show some representative syntax below. The involved defining variables, adding constraints, setting optimization objectives, all these steps are implemented in the main function, referred to as `main_acdcopf`.

1.5.1 Defining variables

In Matlab, Yalmip package is used. Under such a framework, `sdpvar` is used to define the required variables for AC/MTDC OPF, like below

```
var_dc = sdpvar(2*nbuses_dc+8*nconvs_dc, 1);
...
% 1. vn2_dc -the square of dc nodal voltage
vn2_dc = var_dc(1:nbuses_dc, 1);
```

In Matlab, to handle three-dimensional variables where each third-dimension index (k) has different (i, j) dimensions, `cell` is used to store multiple two-dimensional variables, effectively representing a three-dimensional data structure. We apply this rule in defining the variables related to multiple AC power grids, like below:

```
pij_ac = cell(ngrids, 1)
var_ac{ng} = sdpvar(3*nbuses_ac{ng}+2*ngens_ac{ng}, 1);
...
% 6. pij_ac -branch active power flow
pij_ac{ng} = sdpvar(nbuses_ac{ng}, nbuses_ac{ng}, 'full');
```

In Julia, JuMP package is used. Under such a framework, we start with `model = direct_model('')`, then define variables like below

```
var_size_dc = 2*nbuses_dc + 8*nconvs
@variable(model, var_dc[1:var_size_dc])
...
# 1. vn2_dc -the square of dc nodal voltage
vn2_dc = var_dc[1:nbuses_dc]
```

In Julia, to address the issue of defining three-dimensional variables related to multiple AC grids, we use `Vector` to save two-dimension variables like below

```
pij_ac = Vector{Matrix{VariableRef}}(undef, ngrids)
var_size_ac = 3*nbuses_ac[ng] + 2*ngens_ac[ng]
var_ac[ng] = @variable(model, [1:var_size_ac])
...
# 6. pij_ac -branch active power flow
pij_ac[ng] = @variable(model, [1:nbuses_ac[ng], 1:nbuses_ac[ng]])
```

In Python, Pyomo package is used. Under such a framework, we start with `model = ConcreteModel()`, then we define variables like below

```
# 1. vn2_dc -the square of dc nodal voltage
model.vn2_dc = Var(range(nbuses_dc), domain=NonNegativeReals)
```

In Python, to address the issue of defining three-dimensional variables related to multiple AC grids, we directly define like below

```
# 6. pij_ac -branch active power flow
model.pij_ac = Var(
      [(ng, i, j) for ng in range(ngrids) for i in range(nbuses_ac[ng]) for j in range(nbuses_ac[ng])],
      domain=Reals
   )
```

In Cpp, Gurobi Cpp API is used. Under such a framework, the blow code snippet is required at start

```
GRBEnv env = GRBEnv(true);
env.start();
GRBModel model = GRBModel(env);
```

`model.addVar` is used to define the required variables, like below

```
// 1. vn2_dc: the square of DC nodal voltage
Eigen::Matrix<GRBVar, Eigen::Dynamic, 1> vn2_dc(nbuses_dc);
for (int i = 0; i < nbuses_dc; ++i) {
        vn2_dc(i) = model.addVar(-GRB_INFINITY, GRB_INFINITY, 0.0, GRB_CONTINUOUS);
    }
```

In Cpp, to address the issue of defining three-dimensional variables related to multiple AC grids, we use `Eigen::Matrix` to save two-dimension variables like below

```
// 6. pij_ac -branch active power flow
pij_ac[ng] = Eigen::Matrix<GRBVar, Eigen::Dynamic, Eigen::Dynamic>(nbuses_ac[ng], nbuses_ac[ng]);
for (int i = 0; i < nbuses_ac[ng]; ++i) {
    for (int j = 0; j < nbuses_ac[ng]; ++j) {
        pij_ac[ng](i, j) = model.addVar(-GRB_INFINITY, GRB_INFINITY, 0.0, GRB_CONTINUOUS);
       }
   }
```

1.5.2 Adding constraints

In Matlab, constraints are strongly recommended to be written in matrix formulations, as Matlab is highly optimized for matrix operations. Yalmip allows vectorized operations, enabling constraints to be defined in a compact and computationally efficient manner. For example, the soc relaxed MTDC power flow constraints (2.22a)-(2.22d) can be written in the compact matrix form below.

```
% 1. constraints for dc power flow -second-order cone relaxation
zij_dc = 1./abs(ybus_dc); zij_dc = zij_dc-diag(diag(zij_dc));
zij_dc(isinf(zij_dc)) = 1e4;
con_dc = [con_dc; pn_dc == sum(pij_dc, 2) * pol_dc];
con_dc = [con_dc; pij_dc + pij_dc' == zij_dc .* lij_dc];
con_dc = [con_dc; repmat(vn2_dc, 1, nbuses_dc) - repmat(vn2_dc', nbuses_dc, 1) == zij_dc .* (pij_dc - pij_dc')
    ];
con_dc = [con_dc; pij_dc.^2 <= lij_dc .* repmat(vn2_dc, 1, nbuses_dc)];
con_dc = [con_dc; vn2_dc >= 0; lij_dc >= 0];
```

In Julia, JuMP supports vectorized matrix operations and allows constraints to be written compactly. For efficient computation, matrix-based constraints are also recommended, and (2.22a)-(2.22d) can be written below

```
# 1. constraints for dc power flow -second-order cone relaxation
zij_dc = 1.0 ./ abs.(ybus_dc)
zij_dc = zij_dc .- Diagonal(diagm(0 => diag(zij_dc)))
zij_dc[isinf.(zij_dc)] .= 1e4
@constraint(model, pn_dc .== pol_dc * sum(pij_dc, dims=2))
@constraint(model, pij_dc .+ pij_dc' .== zij_dc .* lij_dc)
@constraint(model, vn2_dc .- vn2_dc' .== zij_dc .* (pij_dc .- pij_dc'))
@constraint(model, pij_dc .^ 2 .<= lij_dc .* vn2_dc)
@constraint(model, 0 .<= vn2_dc)
@constraint(model, 0 .<= lij_dc)
```

In Python, Pyomo does not support matrix operations. In this case, constraints like (2.22a)-(2.22d) must be defined using loops like below

```
# 1. constraints for dc power flow -second-order cone relaxation
zij_dc = 1.0 / np.abs(ybus_dc.toarray())
    np.fill_diagonal(zij_dc, 0)
    zij_dc[np.isinf(zij_dc)] = 1e4

```

```python
def nodal_power_flow_dc(model, i):
    return model.pn_dc[i] == pol_dc * sum(model.pij_dc[i, :])
model.nodal_power_flow_dc = Constraint(range(nbuses_dc), rule = nodal_power_flow_dc)

def branch_power_flow_dc(model, i, j):
        return model.pij_dc[i, j] + model.pij_dc[j, i] == zij_dc[i, j] * model.lij_dc[i, j]
model.branch_power_flow_dc = Constraint(range(nbuses_dc), range(nbuses_dc), rule = branch_power_flow_dc)

def voltage_diff_dc(model, i, j):
        return model.vn2_dc[i] - model.vn2_dc[j] == zij_dc[i, j] * (model.pij_dc[i, j] - model.pij_dc[j, i])
model.voltage_diff_dc = Constraint(range(nbuses_dc), range(nbuses_dc), rule = voltage_diff_dc)

def cone_relaxation_dc(model, i, j):
        return model.pij_dc[i, j]**2 <= model.lij_dc[i, j] * model.vn2_dc[i]
model.cone_relaxation_dc = Constraint(range(nbuses_dc), range(nbuses_dc), rule = cone_relaxation_dc)
```

In Cpp, Gurobi Cpp API does not provide built-in matrix operations. We can leverage Eigen library to enhance adding constraints with for-loops like below

```cpp
// 1. Constrains for dc power flow - second-order cone relaxation
Eigen::MatrixXd y_dc_dense = Eigen::MatrixXd(y_dc);
Eigen::MatrixXd zij_dc = y_dc_dense.array().abs().cwiseInverse().matrix();
zij_dc -= zij_dc.diagonal().asDiagonal();
for (int i = 0; i < zij_dc.rows(); ++i) {
     for (int j = 0; j < zij_dc.cols(); ++j) {
         if (std::isinf(zij_dc(i, j))) {
             zij_dc(i, j) = 1e4;
         }
    }
 }

for (int i = 0; i < nbuses_dc; ++i) {
     GRBLinExpr power_flow_sum = 0.0;
     for (int j = 0; j < nbuses_dc; ++j) {
         power_flow_sum += pij_dc(i, j);
    }
     model.addConstr(pn_dc(i) == power_flow_sum * pol_dc);
 }

for (int i = 0; i < nbuses_dc; ++i) {
     for (int j = 0; j < nbuses_dc; ++j) {
         model.addConstr(pij_dc(i, j) + pij_dc(j, i) == zij_dc(i, j) * lij_dc(i, j));
         model.addQConstr(pij_dc(i, j) * pij_dc(i, j) <= lij_dc(i, j) * vn2_dc(i));
         model.addConstr(vn2_dc[i] - vn2_dc[j] == zij_dc(i, j) * (pij_dc(i, j) - pij_dc(j, i)));
    }
 }

for (int i = 0; i < nbuses_dc; ++i) {
     model.addConstr(vn2_dc(i) >= 0.0);
     for (int j = 0; j < nbuses_dc; ++j) {
         model.addConstr(lij_dc(i, j) >= 0.0);
    }
 }
```

1.5.3 Establishing an objective

Establishing an optimization objective is straightforward across different programming environments. The process typically involves two key steps: first, formulating the objective function as a mathematical expression, and then incorporating it into the appropriate solver function. The code snippets regarding constructing an optimization objective in different programming environments are presented below

```matlab
%% "define optimization objectives"
for ng = 1:ngrids
    actgen_ac{ng} = generator_ac{ng}(:, 8);
    if gencost_ac{ng}(1,4) == 3
```

```
        obj_acdc{ng} = sum(actgen_ac{ng}.*( baseMVA^2*gencost_ac{ng}(:,5).*pgen_ac{ng}.^2 + baseMVA*gencost_ac{
            ng}(:, 6).*pgen_ac{ng} + gencost_ac{ng}(:, 7) ));
    elseif gencost_ac(1,4) == 2
        obj_acdc{ng} = sum(actgen_ac{ng}.*( baseMVA*gencost_ac{ng}(:,5).*pgen_ac{ng} + gencost_ac{ng}(:,6) ));
    end
end
for ng = 1:ngrids
    Obj = Obj+obj_acdc{ng};
end
```

```
# "define optimization objectives"
for ng = 1:ngrids
    actgen_ac[ng] = generator_ac[ng][:, 8]
    if gencost_ac[ng][1, 4] == 3
        obj[ng] = sum(actgen_ac[ng] .*
            (baseMVA_ac^2 .* gencost_ac[ng][:, 5] .* pgen_ac[ng].^2 +
            baseMVA_ac .* gencost_ac[ng][:, 6] .* pgen_ac[ng] +
            gencost_ac[ng][:, 7]))
    elseif gencost_ac[ng][1, 4] == 2
        obj[ng] = sum(actgen_ac[ng] .*
                (baseMVA_ac .* gencost_ac[ng][:, 5] .* pgen_ac[ng] +
                gencost_ac[ng][:, 6]))
    end
end
```

```
# "define optimization objectives"
model.objective_terms = []

for ng in range(ngrids):
    actgen_ac = [generator_ac[ng][i, 7] for i in range(ngens_ac[ng])]

    if gencost_ac[ng][0, 3] == 3:
        obj_gencost = sum(
            actgen_ac[i] * (
            baseMVA_ac**2 * gencost_ac[ng][i, 4] * model.pgen_ac[ng, i]**2 +
            baseMVA_ac * gencost_ac[ng][i, 5] * model.pgen_ac[ng, i] +
            gencost_ac[ng][i, 6]
            ) for i in range(ngens_ac[ng])
        )

    elif gencost_ac[ng][0, 3] == 2:
        obj_gencost = sum(
            actgen_ac[i] * (
                baseMVA_ac * gencost_ac[ng][i, 4] * model.pgen_ac[ng, i] +
                gencost_ac[ng][i, 5]
            ) for i in range(ngens_ac[ng])
        )

model.objective_terms.append(obj_gencost)

model.objective = Objective(expr=sum(model.objective_terms), sense=minimize)
```

```
/* DEFINE OPTIMIZATION OBJECTIVE */
GRBQuadExpr obj = 0.0;
for (int ng = 0; ng < ngrids; ++ng) {
    actgen_ac[ng] = generator_ac[ng].col(7);
    if (gencost_ac[ng](0, 3) == 3) {
         for (int i = 0; i < actgen_ac[ng].size(); ++i) {
             obj += actgen_ac[ng](i) * (
                 baseMVA_ac * baseMVA_ac * gencost_ac[ng](i, 4) * (pgen_ac[ng][i] * pgen_ac[ng][i]) +
                 baseMVA_ac * gencost_ac[ng](i, 5) * pgen_ac[ng][i] +
                 gencost_ac[ng](i, 6)
            );
        }
    }
    else if (gencost_ac[ng](0, 3) == 2) {
```

```
         for (int i = 0; i < actgen_ac[ng].size(); ++i) {
             obj += actgen_ac[ng][i] * (
                 baseMVA_ac * gencost_ac[ng](i, 4) * pgen_ac[ng][i] +
                 gencost_ac[ng](i, 5)
            );
        }
    }
}
model.setObjective(obj);
```

1.6 Accessing the Results

By default, the results of the optimization running are printed on the screen, following the well-known Matpower display style, like below

```
==================================================================================
|      AC Grids Bus Data                                                         |
==================================================================================
 Area    Branch       Voltage           Generation              Load
 #       #       Mag [pu]  Ang [deg]   Pg [MW]   Qg [MVAr]   Pd [MW]   Qd [MVAr]
-----    -----   --------  ---------   --------  ---------   -------   ---------
   1       1       1.060      0.000*   167.922     42.694     0.000       0.000
   1       2       1.014      0.000       -          -       20.000      10.000
   1       3       1.018      0.000       -          -       45.000      15.000
   1       4       1.022      0.000     40.000     23.606    40.000       5.000
   1       5       1.000      0.000       -          -       60.000      10.000
   2       1       1.040      0.000*    76.934    103.668     0.000       0.000
   2       2       1.061      0.000    121.407    163.645     0.000       0.000
   2       3       0.993      0.000       -          -        0.000       0.000
   2       4       1.006      0.000       -          -        0.000       0.000
   2       5       1.000      0.000     84.947     -7.451    90.000      30.000
   2       6       0.999      0.000       -          -        0.000       0.000
   2       7       0.992      0.000       -          -      100.000      35.000
   2       8       1.010      0.000       -          -        0.000       0.000
   2       9       0.980      0.000       -          -      125.000      50.000
-----    -----   --------  ---------   --------  ---------   -------   ---------
 The total generation cost is $9363.58/MWh(€8669.98/MWh)

============================================================================================
|     AC Grids Branch Data                                                                 |
============================================================================================
 Area    Branch   From   To        From Branch Flow          To Branch Flow       Branch Loss
 #       #        Bus#   Bus#    Pij [MW]   Qij [MVAr]    Pij [MW]   Qij [MVAr]  Pij_loss [MW]
 ----    ------   -----  -----  ---------  ----------   ----------  ----------    --------
  1       1        1      2      128.979      41.985      -125.705     -32.163       3.274
  1       2        1      3       38.943       6.889       -37.829      -3.549       1.114
  1       3        2      3        7.839      -5.017        -7.788       5.168       0.051
  1       4        2      4        8.023      -6.842        -7.958       7.036       0.065
  1       5        2      5       28.531       2.761       -28.212      -1.804       0.319
  1       6        3      4        0.617     -10.911        -0.606      10.946       0.012
  1       7        4      5        8.564       6.367        -8.477      -6.105       0.087
  2       1        1      4       76.934     103.668       -76.934     -16.397       0.000
  2       2        4      5      -13.730       9.430        13.776      -9.178       0.047
  2       3        5      6       15.009      -2.469       -14.918       2.862       0.090
  2       4        3      6        0.000       0.000         0.000      20.156       0.000
```

2	5	6	7	14.918	5.259	-14.888	-5.005	0.030
2	6	7	8	-85.113	-12.389	85.752	17.804	0.639
2	7	8	2	-121.407	-7.225	121.407	163.645	0.000
2	8	8	9	35.652	12.622	-35.203	-10.364	0.449
2	9	9	4	-89.797	-16.498	90.665	23.876	0.868

The total AC network losses is 7.044 MW .

MTDC Bus Data

Bus DC #	Bus AC #	Area #	DC Voltage Vdc [pu]	DC Power Pdc [MW]	PCC Bus Injection Ps [MW]	Qs [MVAr]	Converter loss Conv_Ploss [MW]
1	2	1	1.008	60.000	61.313	40.000	1.229
2	5	1	1.000	-24.443	-23.311	1.909	1.123
3	5	2	0.998	-35.000	-33.837	-0.003	1.145

The total converter losses is 3.498 MW

MTDC Branch Data

Branch #	From Bus#	To Bus#	From Branch Flow Pij [MW]	To Branch Flow Pij [MW]	Branch Loss Pij_loss [MW]
1	1	2	31.987	-31.725	0.262
2	2	3	7.282	-7.268	0.014
3	1	3	28.013	-27.732	0.282

The total DC network losses is 0.557 MW .

Execution time is 2.120s .

References

[1] B. Kocuk, S. S. Dey, and X. A. Sun, "Strong socp relaxations for the optimal power flow problem," *Operations Research*, vol. 64, no. 6, pp. 1177–1196, 2016.

[2] L. Gan and S. H. Low, "Optimal power flow in direct current networks," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 5614–5619.