Solvers

Which solver to use?

There is no perfect solver. Some solvers (IPOPT, Knitro, etc) will be good in some situations and won't converge in others (even if the problem is feasible).

Results in “V. Rigoni and A. Keane, "An Open-Source Optimal Power Flow Formulation: Integrating Pyomo & OpenDSS in Python", 2020 IEEE Power and Energy Society General Meeting, Montreal, 2020” where obtained using **Knitro**. You can download a trial version of Knitro in https://www.artelys.com/solvers/knitro/

How to select the solver?

In “MAIN_Unbalanced_OPF_RUN.py”, find the following lines:

```
#### Choose a NLP solver  - make sure its installed - some solvers
# IPOPT
optimizer = pyo.SolverFactory('ipopt')
optimizer.options["max_iter"] = 100000
optimizer.options["linear_solver"] = 'mumps'

# KNITRO
#optimizer = pyo.SolverFactory('knitroampl')
#optimizer.options["par_numthreads"] = 5
#optimizer.options["algorithm"] = 0 # Indicates which algorithm to
#optimizer.options["presolve"] = 1 # Determine whether or not to us
```

optimizer = pyo.SolverFactory(‘solver_name’) - *selects the solver*

The following “options” will change depending on which solver you use. You can find the available options on the solver manual (https://coin-or.github.io/Ipopt/OPTIONS.html)

IPOPT (open source)

If you don’t have any licensed solver, IPOPT may be a good option to start using the model. However, have in mind that IPOPT will struggle with some of the constraints (e.g. PV operational limits).

I recommend starting with a simple problem (1 feeder and a low time resolution):

```
""" INPUTS """
# Study feeders - Including all feeders will simulate the
Feeder_names = ['Feeder2'] # ['Feeder1','Feeder2','Feeder
voltagebases = [10,0.4] # in kV line to line

# Simulation time
Time_sim_interval = 60 # [5-1440] - Provided profiles hav
Time_start = 60         # First time step >=5
Time_end = 1440      # Last time step <=1440
```

Disable network and PV operational constraints to see if the problem converges. Then, start adding one by one and see which constraint is the one that IPOPT cannot solve. To disable a constraint, simply write a zero on "if 1==1:"

```
# PV operational limits
if 1==0: # I recommend using a
    def PV_inverter_limit_rule(
        max_P =  PV_Gen_data[pv,
```

You can try relaxing these constraints (A < B * sigma, include sigma in the objective function as a penalization).

How to install IPOPT

If you have Anaconda, simply search for IPOPT on the conda cloud:

https://anaconda.org/search?q=ipopt

and follow the instructions given in “Getting started”