EVALUATIONS OF POWER SYSTEMS BY USING OPEN SOURCE SOFTWARE

B.Phani Ranga Raja, Assistant Professor
Department of Electrical and Electronics Engineering
Usharama College Of Engineering And Technology, Telaprolu, Andhra Pradesh, India

D.J.K.K Gopi Chandoor,Student
Department of Electrical and Electronics Engineering
Usharama College Of Engineering And Technology, Telaprolu, Andhra Pradesh, India

N.V.Narendra ,Student
Department of Electrical and Electronics Engineering
Usharama College Of Engineering And Technology, Telaprolu, Andhra Pradesh, India

A.Pardha saradhi,Student
Department of Electrical and Electronics Engineering
Usharama College Of Engineering And Technology, Telaprolu, Andhra Pradesh, India

Ch.NagaDurga,Student
Department of Electrical and Electronics Engineering
Usharama College Of Engineering And Technology, Telaprolu, Andhra Pradesh, India

Abstract- panda power is new open source power system analysis tool that is well suited for applications with a high degree of automation. This paper introduces panda power and demonstrates a typical work flow of defining and analyzing electric power systems. As an exemplary application of panda power, a simple algorithm that analyses the optimal power flow, short circuit of a distribution system is introduced.. The evaluation of future power system scenarios often requires probabilistic methods with a high demand for automation. This introduces the new open source tool panda power, which focuses on easy to use power system analysis with a high degree of automation.

Keywords – panda power, python

I. INTRODUCTION

The evaluation of future power system scenarios often requires probabilistic methods with a high demand for automation. This paper introduces the new open source tool panda power, which focuses on easy to use power system analysis with a high degree of automation. The development of panda power started as an extension of the widely used power flow solver MATPOWER and its port to python, PYPOWER. In PYPOWER, the electric attributes of the network are defined in a case file in the form of a bus/branch model. The bus/branch model formulation is mathematically very close the power flow, which is why it is easy to generate a nodal admittance matrix or other matrices needed for the power flow calculation.

II. PANDA POWER

2.1 About panda power –

Panda power is a new open source power system analysis tool available under a BSD license. It is implemented in Python, guaranteeing free availability and flexible expansion with other open source libraries. Since Python itself is open source, applications in panda power can be parallelized without any license constraints. This makes it well suited for scientific applications which often rely on high performance computing. Panda power combines the data analysis library pandas and the power flow solver PYPOWER to create an easy to use network calculation program aimed at automation of power system analysis and optimization in distribution and sub-transmission networks.

Panda power comes with an extensive library of electric elements, such as ZIP loads, lines, transformer or switches. Based on these network models, panda power allows carrying out power flow, optimal power flow, state estimation and short circuit calculations as well as topological graph search.

2.2. Data strucures –

Panda power is based on a tabular data structure, where every element type is represented by a table that holds all parameters for a specific element and a result table which contains the element specific results of the different analysis methods. The tabular data structure is based on the Python library pandas. It allows the storage of variables of any data type, so that electrical parameters can be stored together with status variables and meta-data, such as names or descriptions. The tables can be easily expanded and customized by adding new columns without influencing the panda power functionality. All inherent panda's methods can be used to efficiently read write and analyze the network and results data. A panda power network (in the following abbreviated as 'net') is a Python dictionary that holds all information about the network. It includes element and a result tables for each element type, such as lines, transformers, switches etc. Element tables hold all input parameters that are specified by the user, while the result table is used by power flow or optimal power flow functions to store the results. Besides the element tables the net data structure also includes standard type data and network wide parameters like frequency, network name or rated apparent power for the per unit system

2.2. Element models—

The electric models and equivalent circuits, representing the different elements, are described in this Section

2.2.1 Bus: Buses represent the nodes in the network. All other elements are connected to one or more buses. With the element based network model, it is possible to connect multiple loads or generators to the same bus. The rated voltage of the buses defines the voltage levels and constitutes the reference framework for the per unit system

2.2.2 Load: Loads are used to model electric consumption. Panda power includes a ZIP model that allows modeling loads with constant power, constant current or constant impedance.

2.2.3 Generator: Generator elements model power generation units with a fixed active power injection at a fixed voltage magnitude in the power flow calculation. If reactive power limits are defined for the generator, the voltage set point might not always be reached in a power flow calculation.

2.2.4 Shunt: Shunts are network elements that can be used to model capacitor banks or choke coils. Shunts are specified by their power values at rated voltage. The shunt model includes a step model which allows to subdivide the shunt power

2.2.5 Line: The line element is used to model cables or overhead lines with a equivalent circuit. It has two different categories of parameters: parameters that depend on one specific line (e.g. line length or the buses which it connects) and parameters that depend on the type of line which is used (e.g., the impedance and capacity per kilometre or maximal thermal current). Panda power comes with a standard type library that allows the creation of lines using the element specific parameters and loading the type specific parameters from a standard type data base

2.2.6 Switch: The switch element allows modeling of ideal switches. It supports switches between two buses or between a bus and a branch element (line or transformer).

2.3 Network Analysis—

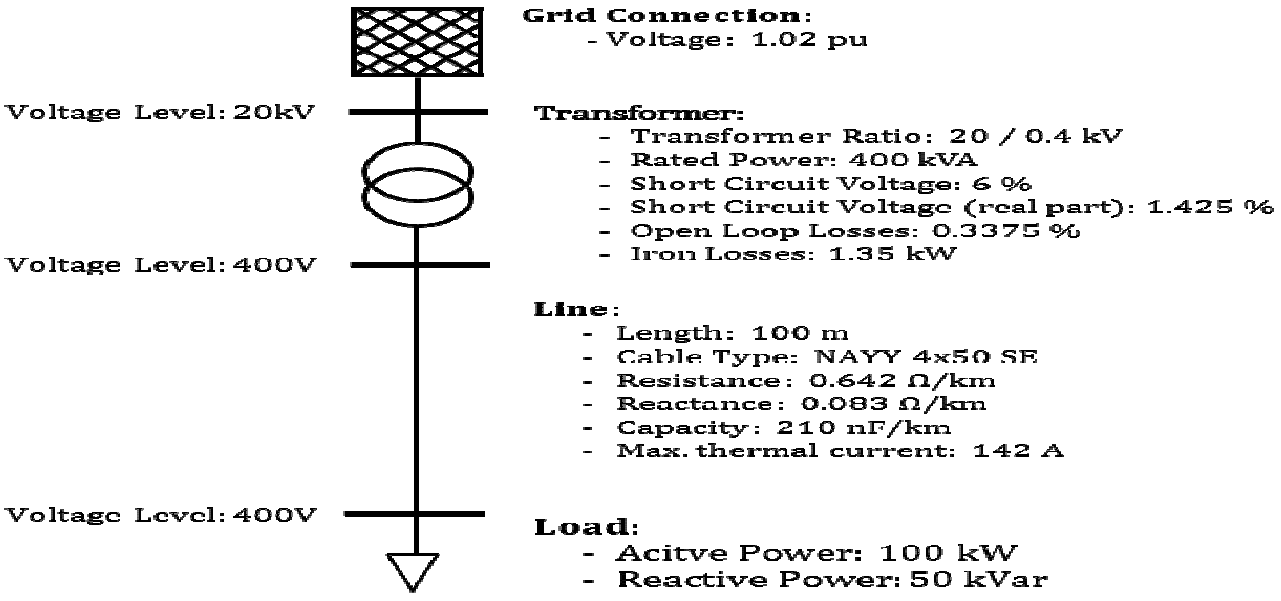
2.3.1 Power Flow: The panda power flow algorithm is originally based on PYPOWER, but has been improved with respect to robustness, runtime and usability. The Newton-Raphson solver [5] is accelerated with just in- time compilation trough the Python library numba. The power flow can be initialized from a DC power flow or from previous results to improve convergence. A connectivity check allows the power flow to converge even if some areas are not connected to an external grid.

2.3.2 Short Circuit Calculation: The short circuit module allows calculation of short-circuit currents according to IEC 60909. The correction factors and calculation rules specified in the standard are applied automatically when carrying out a short circuit calculation. The module allows calculating symmetrical three-phase and unsymmetrical two-phase short circuit currents

2.3.3 State Estimation: panda power includes an implementation of a state estimation with weighted-least-square approach. The state estimation module supports voltage, active power and reactive power measurements at buses, lines and transformers. It includes a bad data detection based on a _2 test and a normalized residual test that allows detecting and removing bad measurements.

2.4 Experiment Results

To demonstrate the pandapower workflow a minimal example is presented in this section. The example shows how networks are created and analyzed using the pandapower API



.

```
import pandapower as pp

#create empty net
net = pp.create_empty_network()

#create buses
bus1 = pp.create_bus(net, vn_kv=20., name="Bus 1")
bus2 = pp.create_bus(net, vn_kv=0.4, name="Bus 2")
bus3 = pp.create_bus(net, vn_kv=0.4, name="Bus 3")

#create bus elements
pp.create_ext_grid(net, bus=bus1, vm_pu=1.02)
pp.create_load(net, bus=bus3, p_kw=100, q_kvar=50)

#create branch elements
trafo = pp.create_transformer(net, hv_bus=bus1, lv_bus=bus2,
                              std_type="0.4 MVA 20/0.4 kV")
line = pp.create_line(net, from_bus=bus2, to_bus=bus3,
                      length_km=0.1, std_type="NAYY 4x50 SE")
```

Running a Power Flow—

For the created network panda power's runpp function calculates the power flow and stores the results into the result tables with the prefix "res " for each element (e.g., res bus, res line ...). Figure 3 shows how the results can be extracted and the respective console output.

```
pp.runpp(net)
print("Trafo loading: %.2f %%"%net.res_trafo.loading_percent.at[0])
print("Line loading: %.3f %%"%net.res_line.loading_percent.at[0])
print("Voltage vector: %s"%net.res_bus.vm_pu.values)
```

```
Trafo loading: 29.29 %
Line loading: 117.835 %
Voltage vector: [ 1.02        1.00884272  0.96443057]
```

Running a Short Circuit Calculation—

To run a short circuit calculation, the short circuit power and r/x ratio of the external grid have to be specified as additional parameters compared to the power flow calculation. Here, the maximum short circuit currents for faults at each bus are calculated. The correction factors for the voltage source and transformer are automatically applied according to the IEC 60909 standard. The calculation returns current values for initial short circuit current ikss_ka and peak short circuit current ip_ka

```
import pandapower.shortcircuit as sc
net.ext_grid["s_sc_max_mva"] = 100
net.ext_grid["rx_max"] = 0.1
sc.calc_sc(net, case="max", ip=True, r_fault_ohm=2.)
print(net.res_bus_sc)
```

```
     ikss_ka     ip_ka
0   2.534707  4.317318
1   0.126631  0.182666
2   0.122698  0.176991
```

IV.CONCLUSION

Panda power demonstrated how it is well suited to tackle tasks evaluating future scenarios in distribution systems with a high degree of automation. Minimal examples that show the typical work flow with panda power have been presented. Tools such as panda power are therefore necessary to allow automated investigations with a high degree of automation for a large number of grids

REFERENCES

[1] O. H. Jadhav and S. M. Jadav, “Aspects of renewable energy potentia in india and future scope,” in 2017 International Conference o Nascentt Technologies in Engineering (ICNTE), Jan 2017, pp. 1–6.

[2] “Zeitreihen zur entwicklung der erneuerbaren energien in deutsch-land,” Federal Ministry for Economic Affairs and Energy,Germany,2017 (in German).

[3] W. McKinney, “pandas: a foundational python library for data analysi and statistics,” Python for High Performance and Scientific Computing, pp. 1–9, 2011.

[4] “IEEE recommended practice for industrial and commercial powe systems analysis (brown book),” IEEE Std 399-1997, pp. 1–488, Aug 1998.

[5] J. J. Grainger and W. D. Stevenson, Power system analysis. McGraw-Hill, 1994.

[6] H. Wang, C. E. Murillo-Sanchez, R. D. Zimmerman, and R. J.Thomas, “On computational issues of market-based optimal powerflow,” IEEE Transactions on Power Systems, vol. 22, no. 3, pp. 1185–1193, 2007.

[7] “IEC 60909-0:2016: Short-circuit currents in three-phase a.c. systems- part 0: Calculation of currents,” International Standard, 2016