OpenDSS Training Workshop - 2021

Advanced topics & Applying DSS in R&D

Jeremiah Deboever
Paulo Radatz
EPRI Knoxville, TN

02 September 2021

Instructors

▪ Jeremiah Deboever

Jeremiah is a Research/Scientist III with EPRI in Palo Alto, California USA. He holds a PhD in Electrical Engineering from Georgia Institute of Technology, Atlanta GA, a MS in Renewable Energy Engineering from Oregon Institute of Technology, Wilsonville OR, and a BS in Mechanical Engineering from Binghamton University, Binghamton NY. Jeremiah has worked on different projects around distribution system modeling and simulation with a focus on electric vehicle impact studies, DER integration studies, hosting capacity analysis, distribution load modeling, and quasi-static time-series simulation. He has published over 20 publications on electric distribution modeling/simulation and won a best poster award at the World Conference on PV Energy Conversion in 2018.

▪ Paulo Radatz

Paulo Radatz serves as Engineer/Scientist II at the Electric Power Research Institute (EPRI) in Knoxville, Tennessee, USA. He received both his Masters and Bachelors's degree in electrical engineering, emphasizing energy and automation, from the University of Sao Paulo, Sao Paulo, Brazil. He was awarded a prize for being the best bachelor's student of Polytechnic School of University of Sao Paulo (2015). He has six years of experience with OpenDSS, having taught several OpenDSS trainings in Brazil at conferences, universities, and industry. He is the founder of the largest YouTube channel about OpenDSS in the world.

Agenda

OpenDSS in R&D

Example #1: Time-series Simulation

Example #2: Hosting Capacity Analysis

Closing Remarks

OpenDSS in R&D

Outline

Available tools:

- OpenDSS
- OpenDSS-G
- Excel VBA
- MATLAB toolbox
- Python toolbox

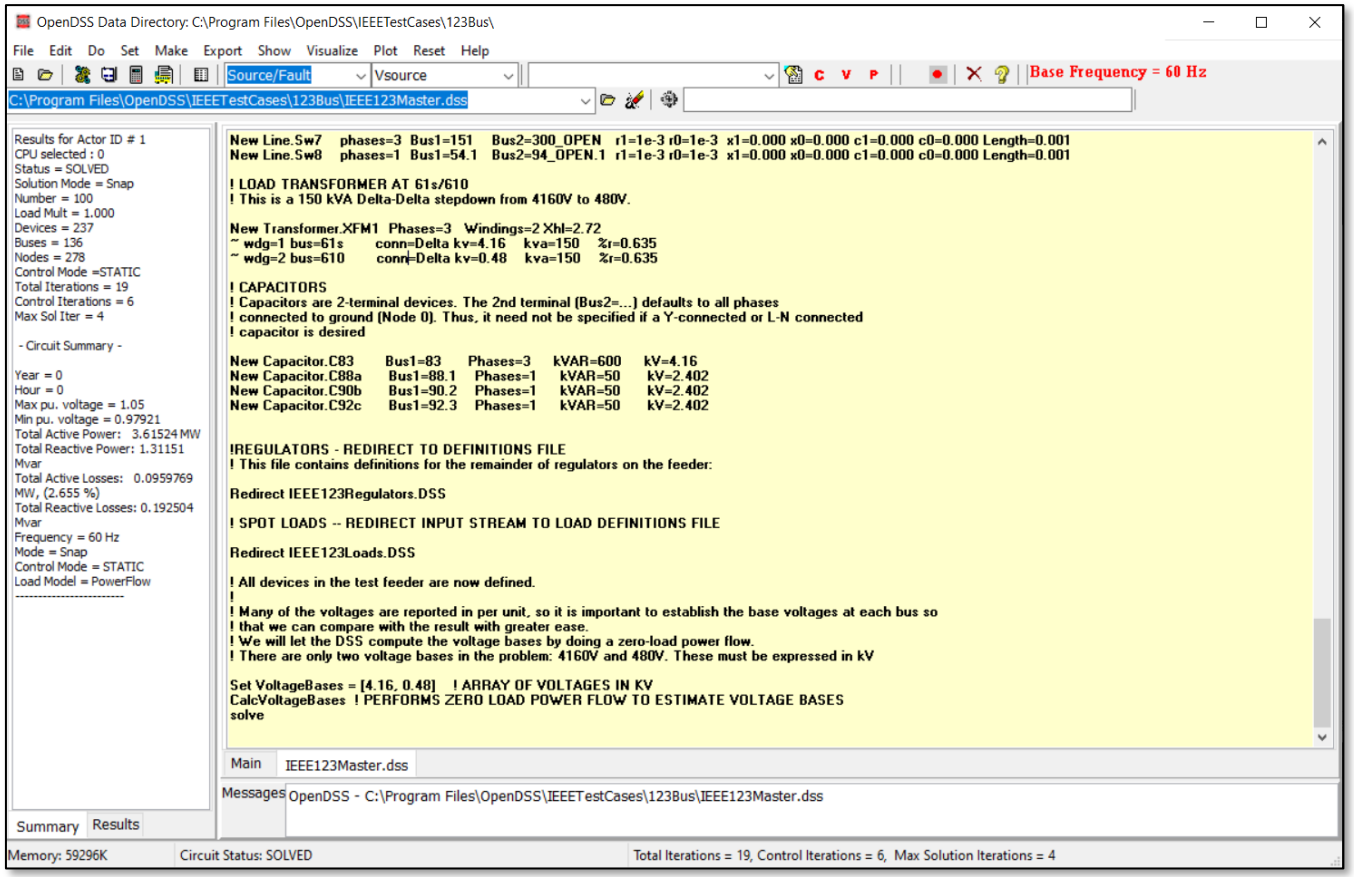
Available feeder models:

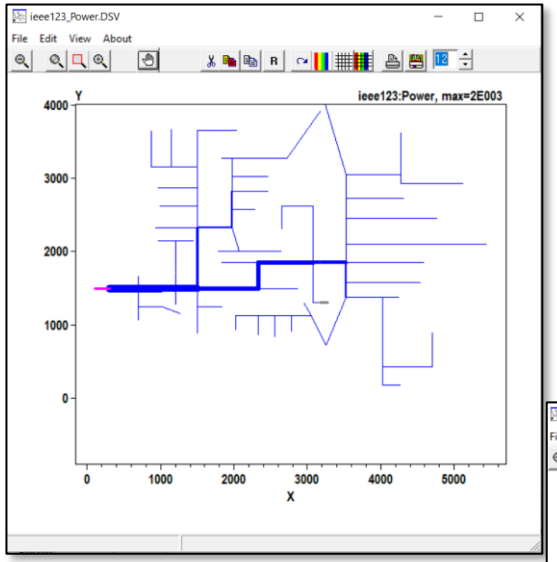
- IEEE models
- EPRI models

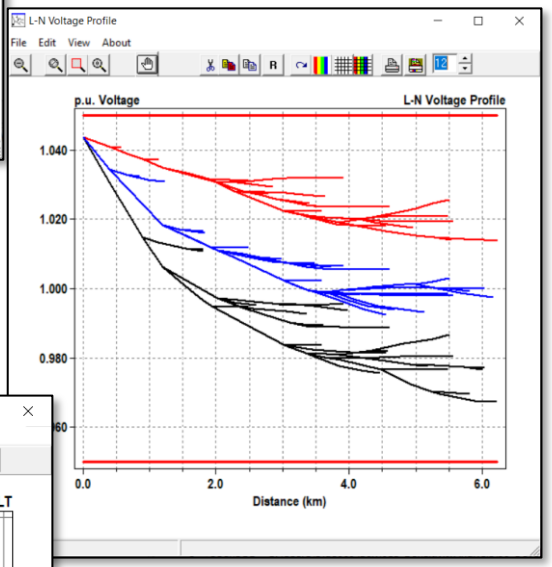
Available datasets:

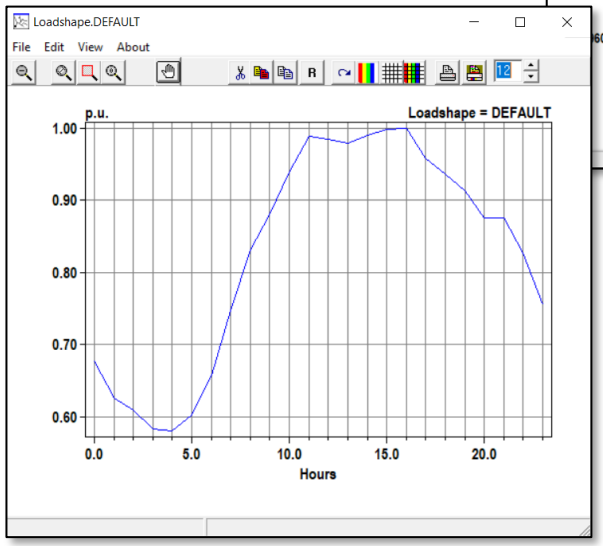
- Load / PV profiles
- 3rd party datasets

Scripts in OpenDSS





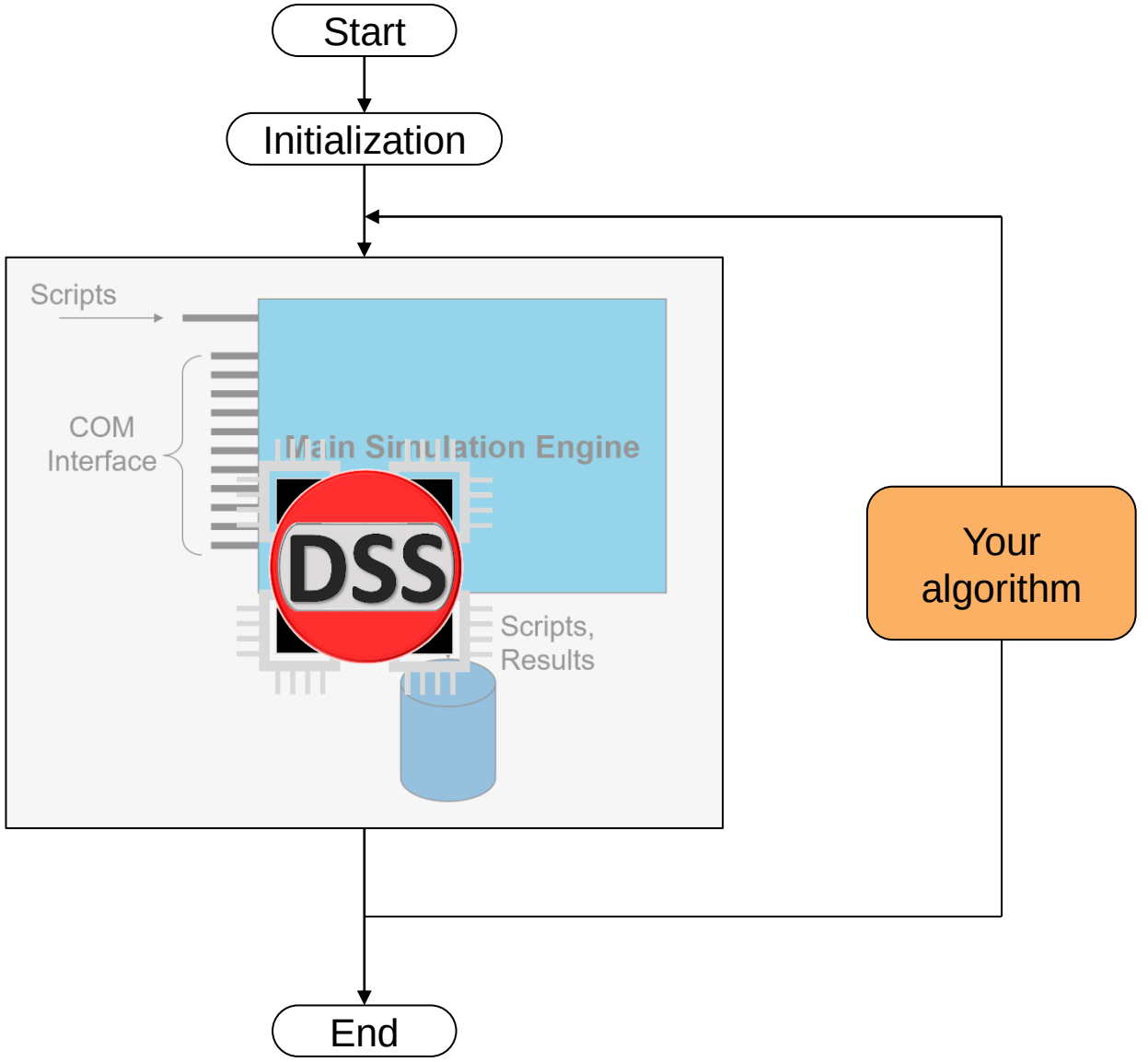




OpenDSS-G

See session #2 from this virtual training

Using OpenDSS for its simulation engine



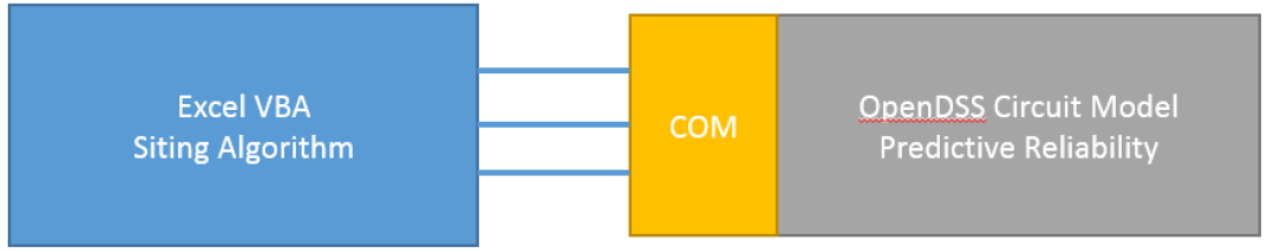
Volt-var optimization (VVO)

Energy storage logic

Conservation voltage reduction (CVR)

Distributed Energt Resource Management Systems (DERMS)

Excel VBA



Start DSS

Specify DSS master file

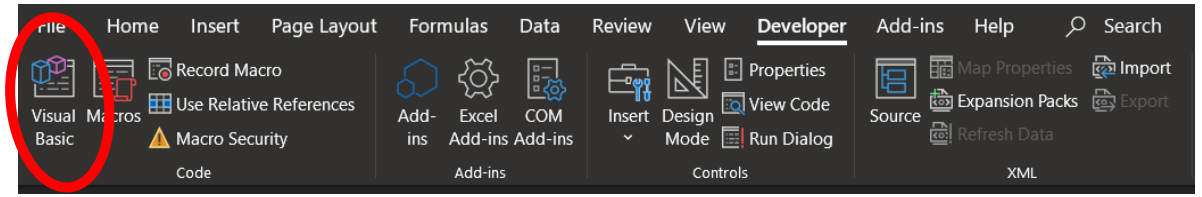
Execute commands

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1															
2		Start DSS	Version: Version 8.6.1.1 (32-bit build); License Status: Open					C:\Users\prdu001\OpenDSS\Test\SimpleStorageTest.dss							
3								C:\Users\prdu001\DSSData\TEMCgenmodels\Model_5\Master.DSS							
4								C:\Users\prdu001\OpenDSS\Test\PVSystemTest-Duty.dss							
5															
6		***Run File Path Name:***	C:\Users\prdu001\OpenDSS\Distrib\IEEETestCases\8500-Node\Run_8500Node.dss					[C:\Users\prdu001\DSSData\BenYork\FRT12 Model_031516\FRT12_CKT.dss]							
7															
8			Execute the Run File	Edit		C:\Users\prdu001\OpenDSS\Distrib\IEEETestCases\123Bus\Run_MonteCarloFault.dss									
9						C:\Users\prdu001\OpenDSS\Distrib\IEEETestCases\8500-Node\Run_8500Node.dss									
10			Solve the Circuit			C:\Users\prdu001\DSSData\Wes\Colfax\Run_Master_Storage_50.DSS									
11						C:\Users\prdu001\OpenDSS\Distrib\IEEETestCases\13Bus\IEEE13test.dss									
12						C:\Users\prdu001\OpenDSS\Distrib\EPRITestCircuits\ckt5\Run_ckt5.dss									
13						C:\Users\prdu001\OpenDSS\Distrib\IEEETestCases\13Bus\IEEE13Nodeckt.dss									
14						(C:\users\prdu001\OpenDSS\distrib\IEEETestCases\123Bus\IEEE123Master.DSS)									
15		Load the Seq Voltages	Load the a + jb Volt Sheet	Load the Mag/Angle Sheet		C:\Users\prdu001\Projects\EPRI\GIC\Harmonics2018\OpenDSSDemo\TestCircuitHarrn500kV-2.dss									
16															
17		Load the Seq Currents	Load the Load Powers			C:\Users\prdu001\OpenDSS\Distrib\EPRITestCircuits\ckt24\master_ckt24.dss									
18															
19															
20						Export mon TRYD									
21															
22			Execute Selected Commands			New Energymeter.M1 Line.L115 1									
23						Energymeter.m1.action=z									
24															
25		***Result:***													
26								show buses							
27		**Some Common Commands:**						Export voltages							
28															
29			Set MarkRecl=yes Markrelay=yes	clear				Get datapath							
30			? Energymeter.m1.saifi	show event				plot zone							
31			Plot Circuit	show overload				plot profile							
32			show v ln nodes	show element capcontrol				Export Sections							
33			show curr elem					FileEdit @Lastfile							

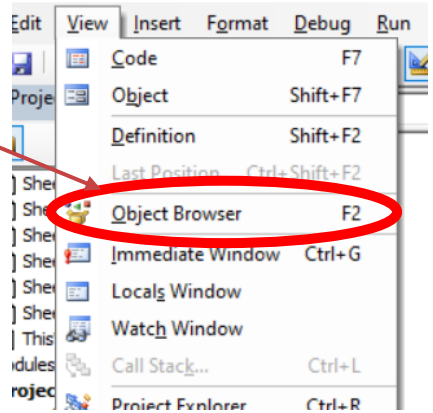
Example: "C:\Program Files\OpenDSS\Examples\Excel\DSSDriver8.xlsm"

Excel VBA – Object Browser

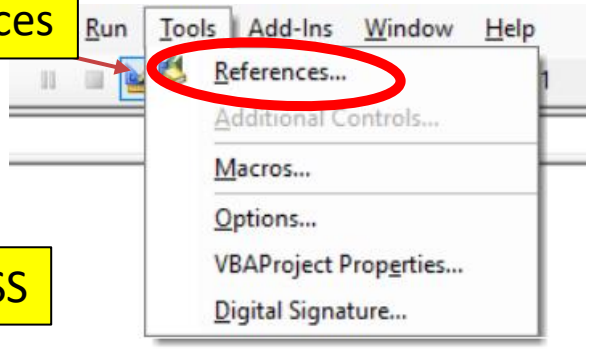
Open VBA



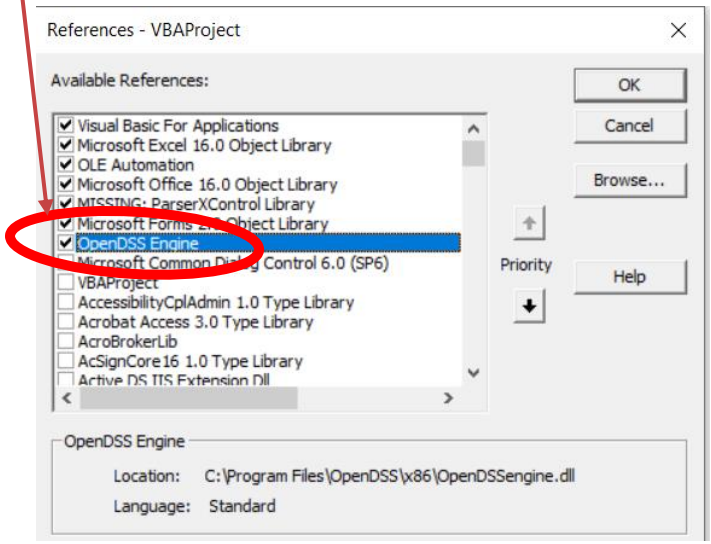
Open Object Browser



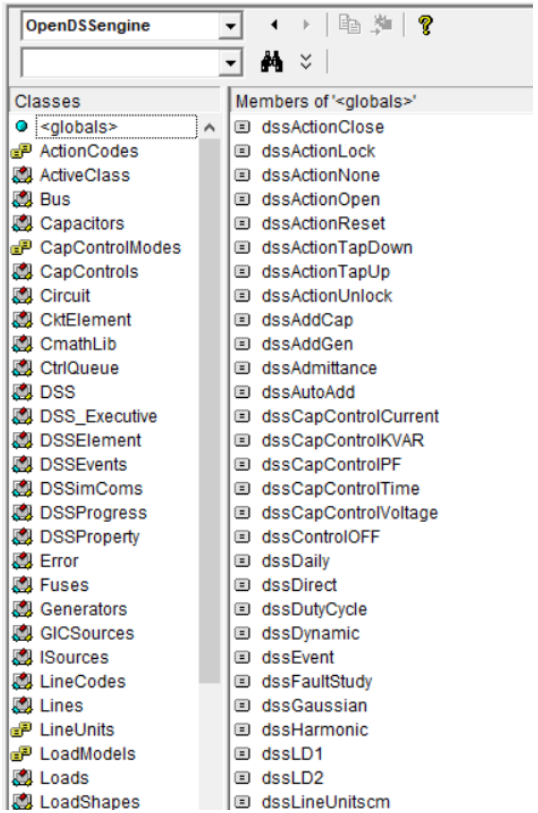
Select References



Select OpenDSS

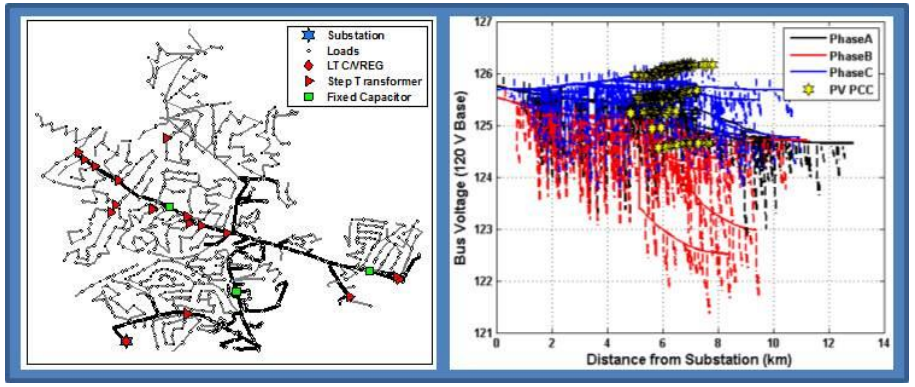


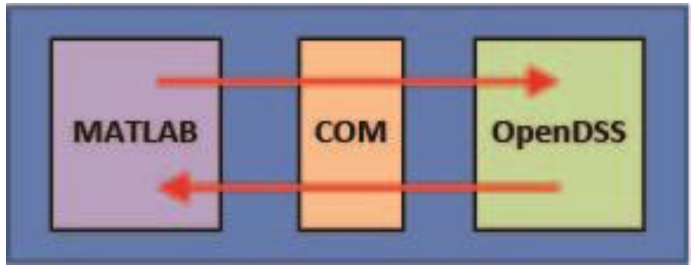
Object Browser

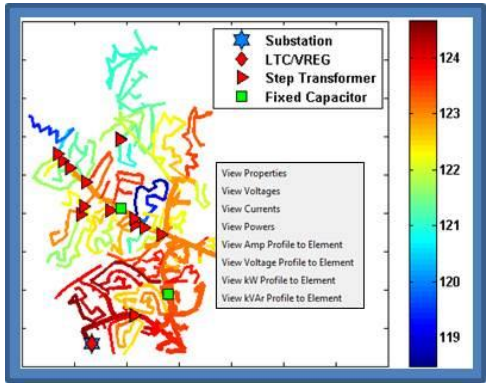


MATLAB integration: GridPV Toolbox

- Set of MATLAB function created by Sandia National Laboratory
- To model and simulate the integration of distributed generation into the electric power system and to determine the impacts on the distribution system for highly variable generation
- Available at: https://pvpmc.sandia.gov/applications/gridpv-toolbox/

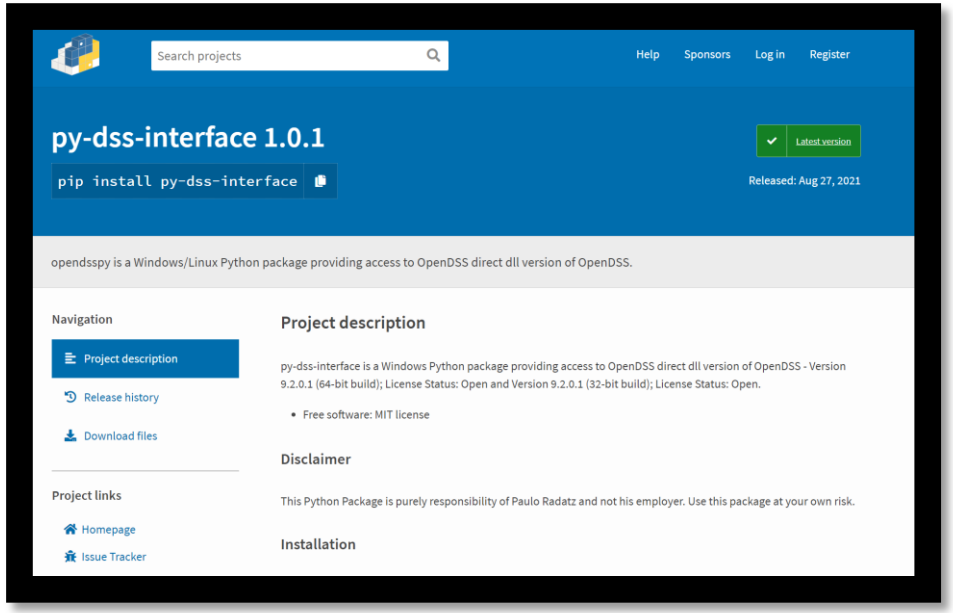






Python integration: py-dss-interface

- Python Package
- Uses the official OpenDSS DirectDLL version
  - Version 9.3.0.1 comes with the Package
  - Users can use their OpenDSS version as well
- Created based on Direct connection Shared Library (DLL) for OpenDSS doc by Davis Montenegro



- Installing from PyPI
  - *pip install py-dss-interface*

Getting Started

User can pass the OpenDSS path as argument: *"C:/Program Files/OpenDSS"*

import package

Create dss Obj

Use text method of dss

Use solution_solve method of dss

Use circuit_all_bus_volts method of dss

```python
# First import the Package
import py_dss_interface

# Creates an OpenDSS object
dss = py_dss_interface.DSSDLL()

# Select the DSS model
dss_file = r"C:\Program Files\OpenDSS\IEEETestCases\13Bus\IEEE13Nodeckt.dss"

# Compile
dss.text(f"compile [{dss_file}]")

# Solve
dss.solution_solve()

# Show Voltage Report
dss.text("show voltages")

# Get all buses voltages
all_bus_volts = dss.circuit_all_bus_volts()

print(all_bus_volts)
```

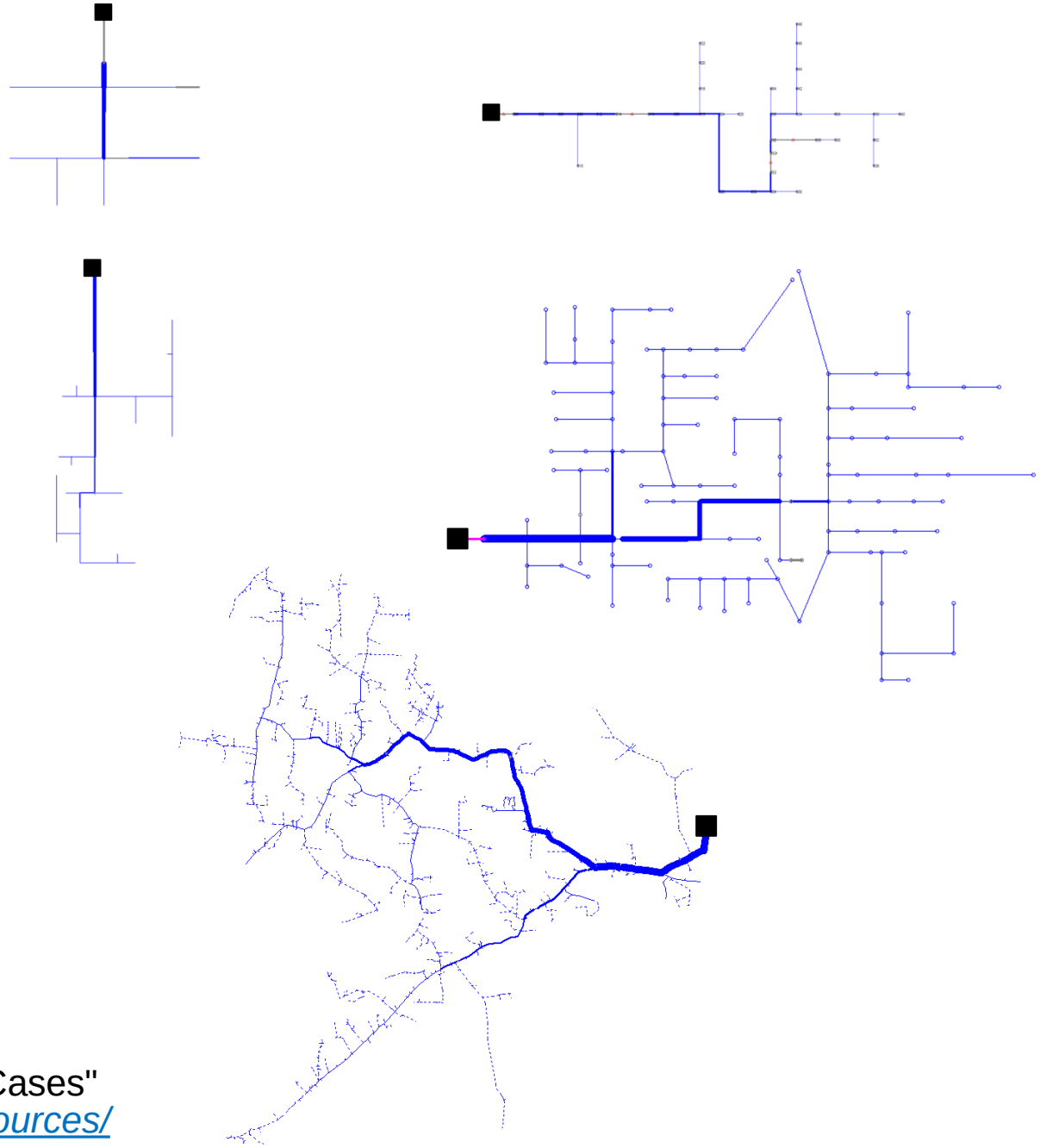
IDE Code Completion is quite useful to select methods of dss

Know more about py-dss-interface

- Resources
  - PyPI (Python Package Index): https://pypi.org/project/py-dss-interface/
  - GitHub: https://github.com/PauloRadatz/py_dss_interface
  - Doc: https://py-dss-interface.readthedocs.io/en/latest/usage.html
  - OpenDSS_Direct_DLL.pdf:https://sourceforge.net/p/electricdss/code/HEAD/tree/trunk/Version8/Distrib/Doc/OpenDSS_Direct_DLL.pdf
  - YouTube Videos: https://www.youtube.com/playlist?list=PLhdRxvt3nJ8zlzp6b_-7s3_YwwlunTNRC

IEEE test feeders

- 13-bus feeder: simple feeder model to test common features
- 34-bus feeder: Actual feeder located in Arizona, which is highly loaded.
- 37-bus feeder: Actual feeder in California with delta configuration.
- 123-bus feeder: Test feeder with significant voltage drop requiring voltage regulation.
- 8500-node feeder: Large scale feeder intended to test the scalability of an algorithm. Secondaries are modeled.

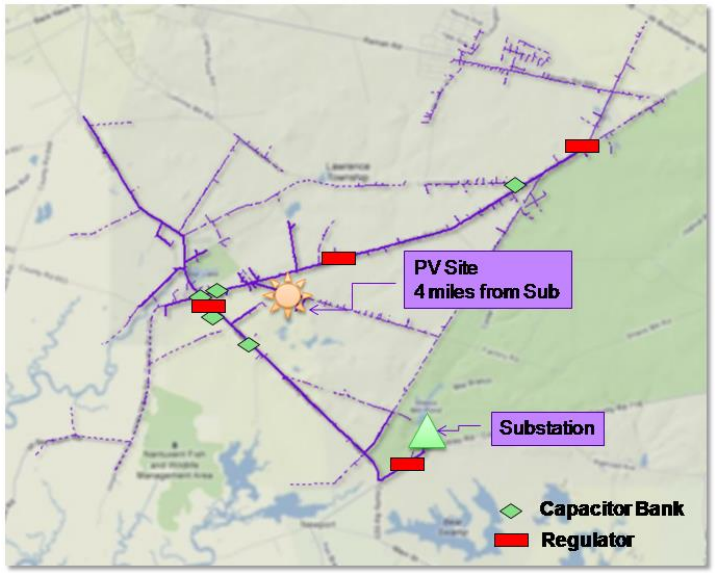


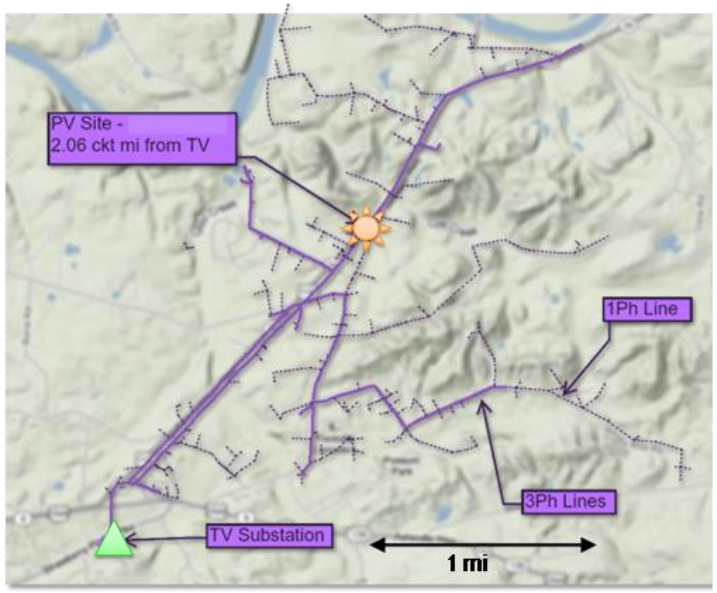
"C:\Program Files\OpenDSS\IEEETestCases"

Available at: https://site.ieee.org/pes-testfeeders/resources/

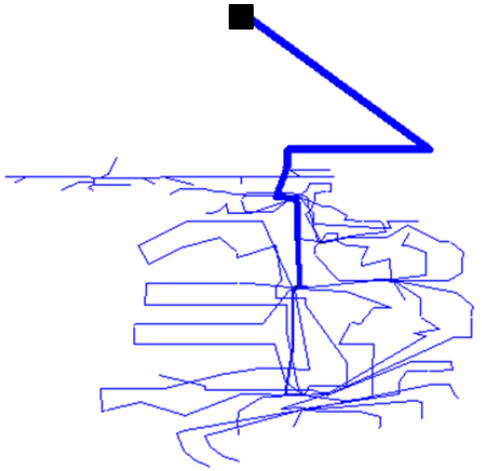
EPRI test feeders

- Feeder J1: Located in northeastern US feeder with 1.7MW of customer-owned PV systems.





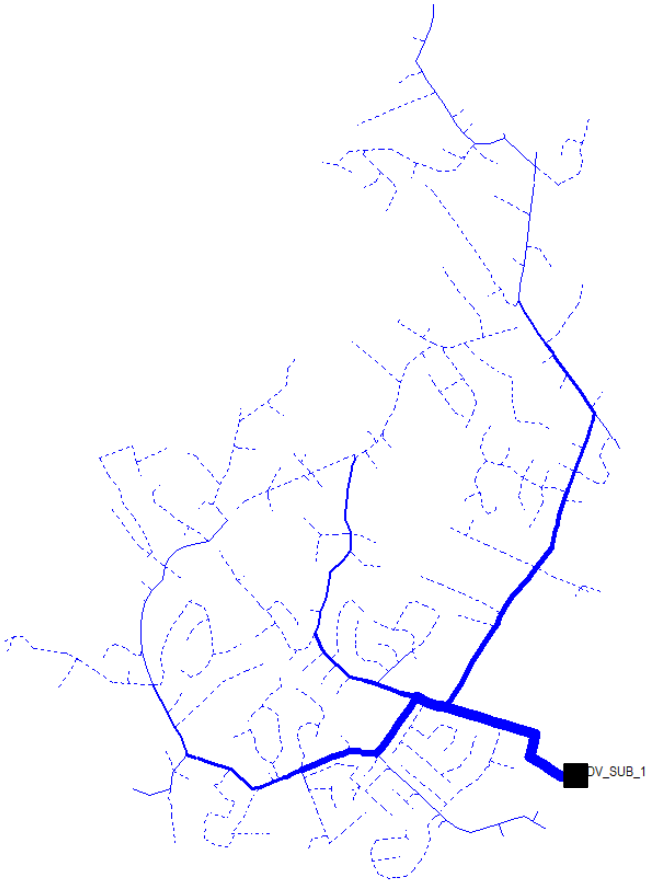
- Feeder K1: Located in southeastern US with a 1.0MW customer-owned PV system.



- Feeder M1: Short and compact feeder with 1500 residential customers and radio-controlled capacitor banks.

Available at: https://dpv.epri.com/index.html

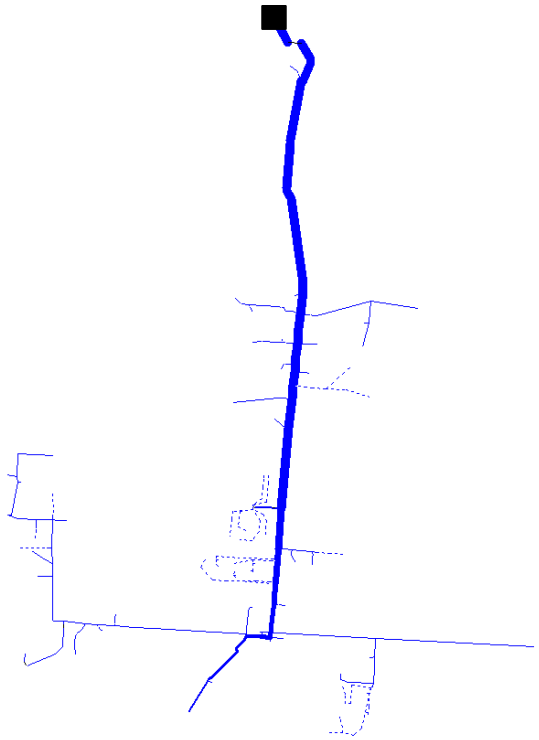
EPRI test feeders (cont'd)



EPRI Ckt5

Primary voltage: 12.5kV

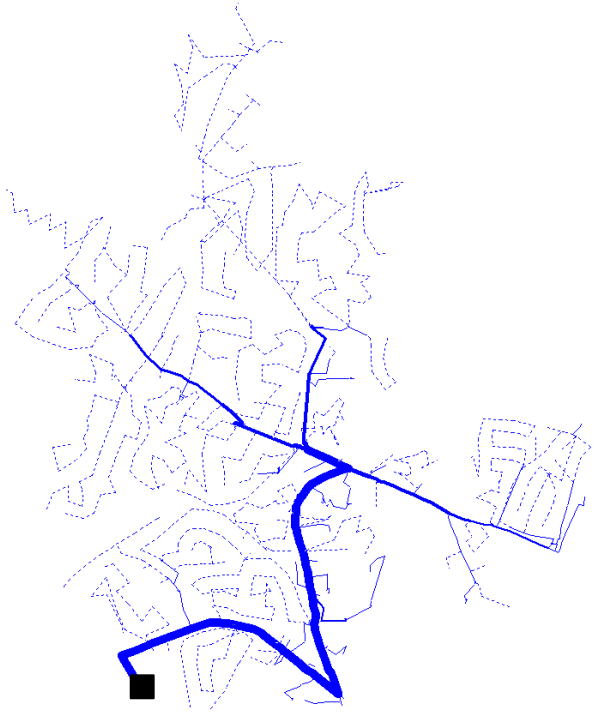
# of feeders: 1



EPRI Ckt7

Primary voltage: 12.5kV

# of feeders: 14



EPRI Ckt24

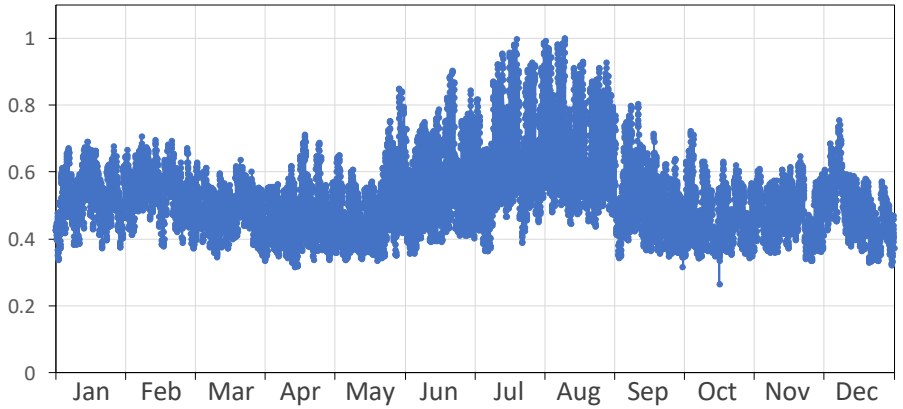
Primary voltage: 34.5kV

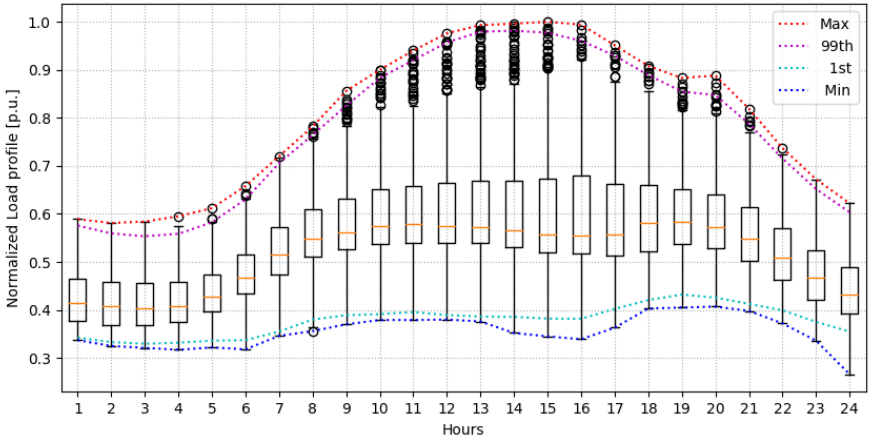
# of feeders: 2

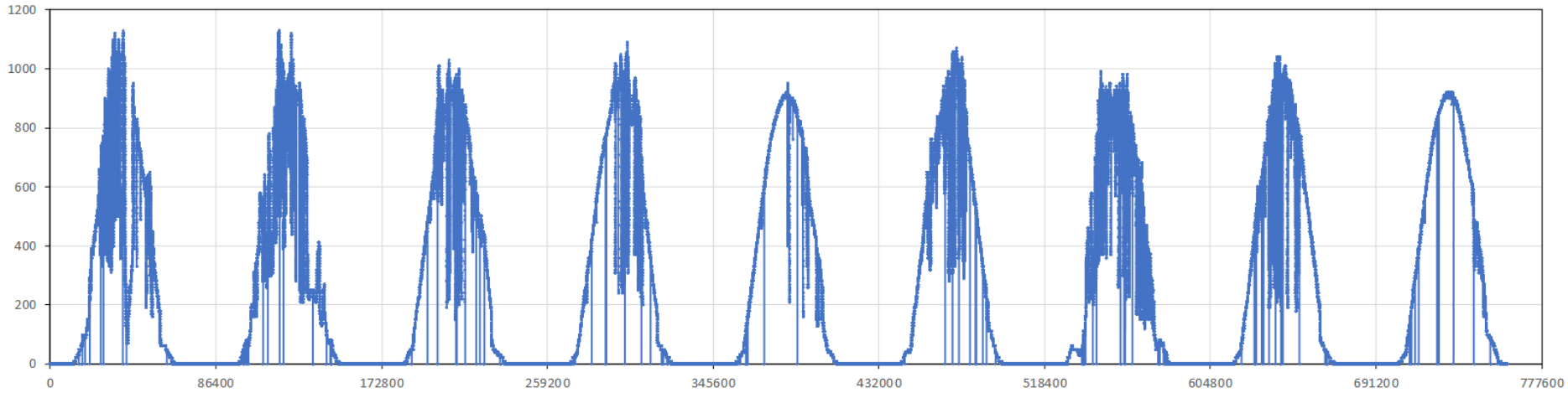
Available at: "C:\Program Files\OpenDSS\EPRITestCircuits"

Datasets

- Load profiles: (available with feeder model)
  - IEEE 123 bus: ‘paperloadshape.txt’
  - EPRI ckt5: ‘loadshape_ckt5.dss’
    - Residential load profile
    - Small & Medium commercial load profiles
- PV profiles:
  - EPRI models: (available at: dpv.epri.com/)
    - Resolutions: 1 sec, 1-, 15-, 60-min
  - NREL dataset: (available at: pvwatts.nrel.gov/)
    - Location-specific TMY data at 1hr resolution

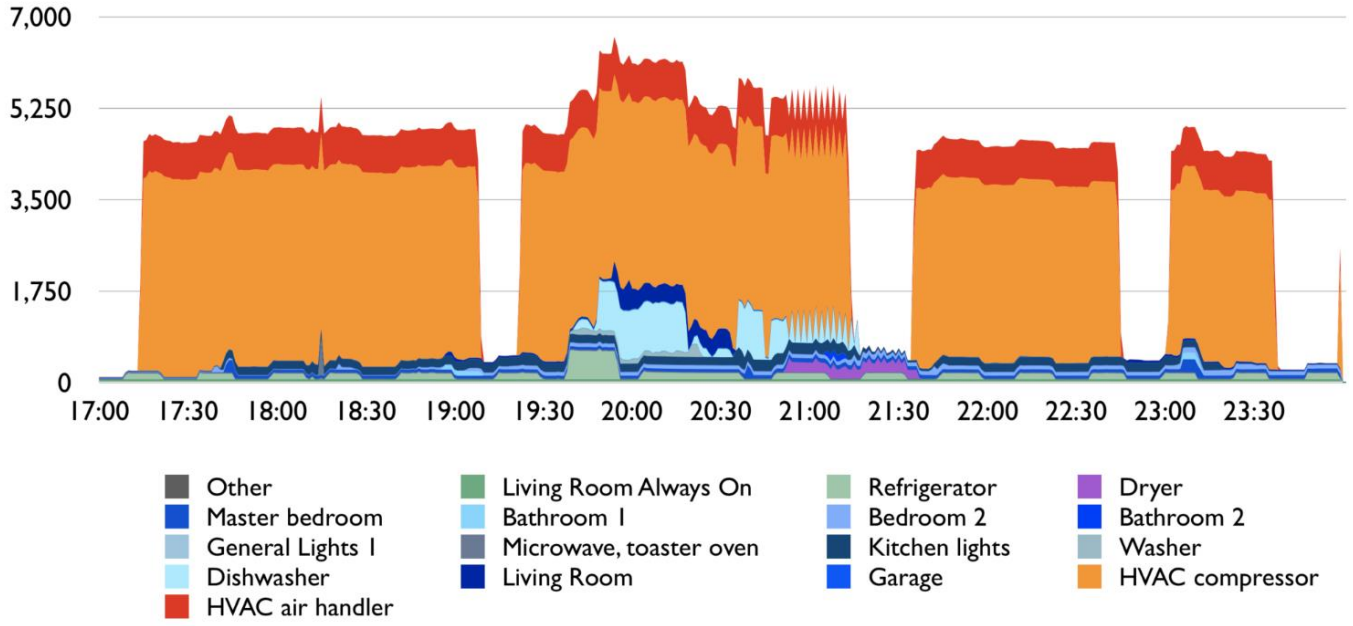






3rd party datasets: Pecan Street

- Available at: https://www.pecanstreet.org/dataport/about/
- Dataport provides access to a variety of residential energy and water data including:
  - Whole-home and circuit-level energy use,
  - Appliance use,
  - Solar generation,
  - EV charging behavior,
  - HVAC use, etc.

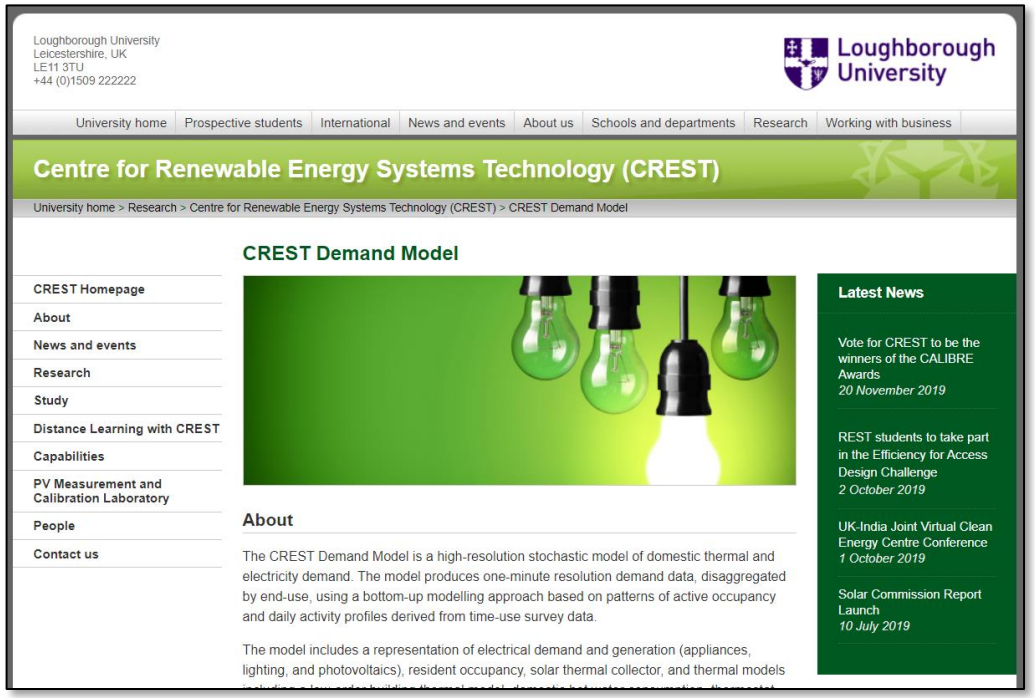


https://www.pecanstreet.org/wp-content/uploads/2019/01/1-minute.png

3rd party datasets: Centre for Renewable Energy Systems Technology (CREST)

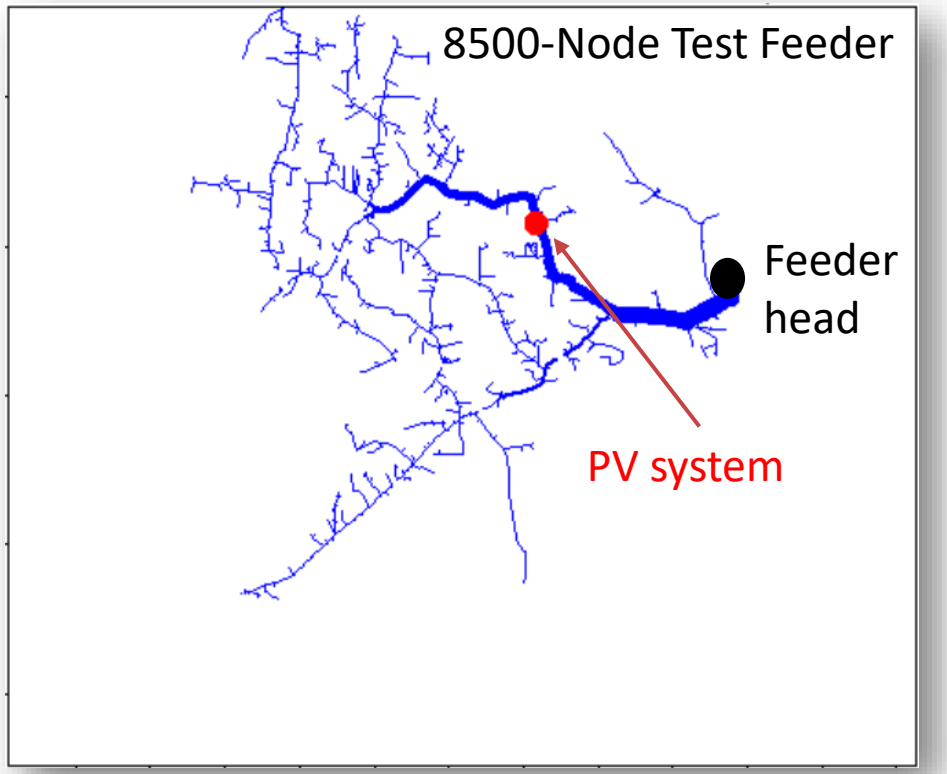
- Available at: https://www.lboro.ac.uk/research/crest/demand-model/

- High-resolution stochastic model of domestic thermal and electricity demand.
- Produces one-minute resolution demand data, disaggregated by end-use.
- Using a bottom-up modelling approach based on patterns of active occupancy and daily activity profiles



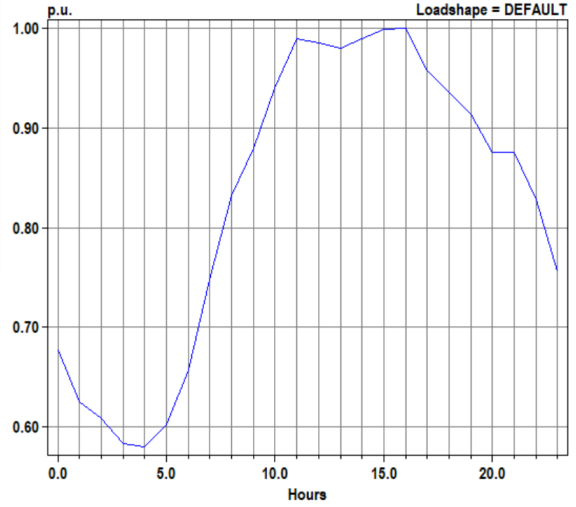
Example #1: Time-series Simulation

Feeder

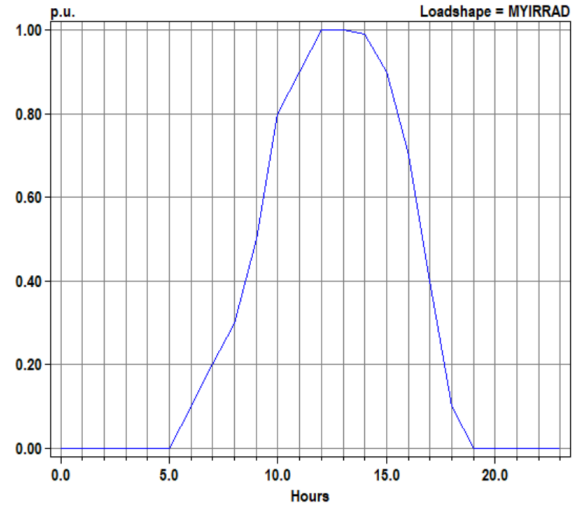


- Voltage regulating devices
  - Regulators
  - Capacitors
- PV system
  - kW rating = 8000
  - kVA vs kW ratings = 1
  - DC-to-AC ratio = 1
  - Reactive power priority

Loadshape applied for all loads



PV system's irradiance



8500-Node Test Feeder

Study Objectives

Evaluate the impact of the smart inverter functions on the feeder energy metrics, such as feeder consumption, losses, and PV system generation.

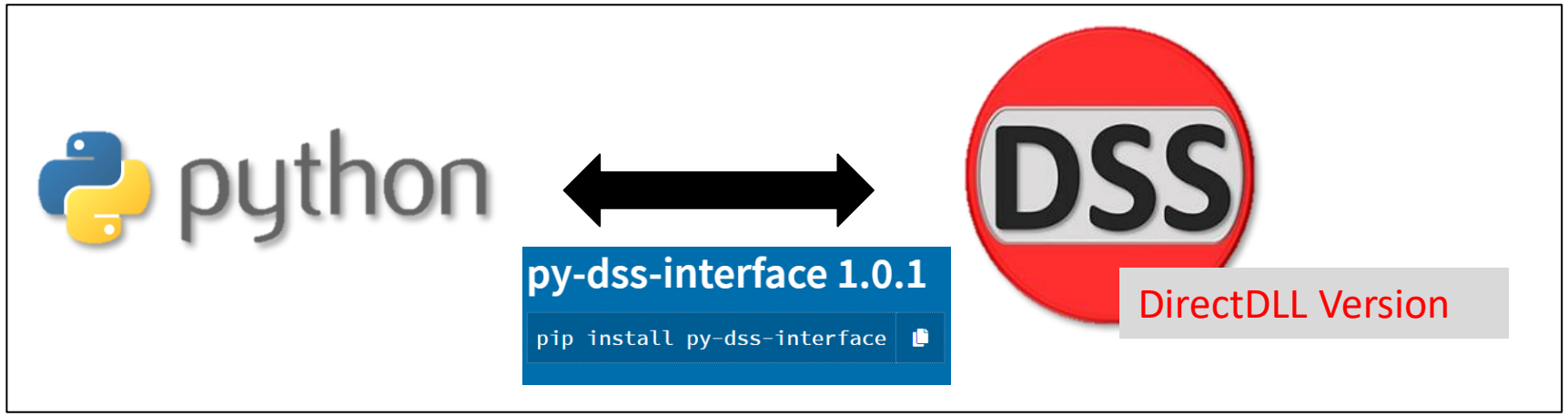
Simulation

- Perform a daily time-series analysis with timestep of 1h considering 3 PV System smart inverter function scenarios:
  - Unity-power factor
  - 0.9 absorbing constant power factor
  - Volt-var IEEE1547-2018 catB
- Obtain the following metric results
  - Feeder head kWh and kvarh
  - Feeder energy losses kWh
  - Total load kWh
  - PV system generation kWh

Data Analysis

- Compare the metric results for the 3 scenarios
- For simplicity, the comparison can be a simple bar plot for each metric

Simulation Part



- Process for each smart inverter function
  1. Compile Circuit
  2. Set baseline
     1. Add energymeter
     2. Define default loadshape to all loads
     3. Change solution maxiterations and maxcontroli property values
  3. Set time-series solution properties
  4. Add PV system and the smart inverter function
  5. Run the simulation process (Solve command)
  6. Read Energymeter results
  7. Save results in a csv file

Process (process.py)

Smart Inverter Function options

Master.dss path

Create dss Object

PV system bus

```
40  smart_inverter_functions_list = ["unity-pf", "pf", "volt-var"]
41
42  script_path = os.path.dirname(os.path.abspath(__file__))
43  dss_file = pathlib.Path(script_path).joinpath("Feeders", "8500-Node", "Master.dss")
44
45  dss = py_dss_interface.DSSDLL()
46
47  bus = "l3104830"
48
49  feeder_kwh_list = list()
50  feeder_kvarh_list = list()
51  loads_kwh_list = list()
52  losses_kwh_list = list()
53  pv_kwh_list = list()
```

"...\TimesSeries_OpenDSS\Feeders\8500-Node\Master.dss"

Process (process.py)

```python
for smart_inverter_function in smart_inverter_functions_list:
    # Process for each smart inverter function
    dss.text(f"Compile [{dss_file}]")
    set_baseline()
    set_time_series_simulation()

    # Add PV system and the smart inverter function
    SmartInverterFunction(dss, bus, 12.47, 8000, 8000, smart_inverter_function)

    dss.text("solve")

    # Read Energymeter results
    energymeter_results = get_energymeter_results()
    feeder_kwh_list.append(energymeter_results[0])
    feeder_kvarh_list.append(energymeter_results[1])
    loads_kwh_list.append(energymeter_results[2])
    losses_kwh_list.append(energymeter_results[3])
    pv_kwh_list.append(energymeter_results[4])
```

Process (process.py)

```
55  for smart_inverter_function in smart_inverter_functions_list:
56      # Process for each smart inverter function
57      dss.text(f"Compile [{dss_file}]")
58      set_baseline()
59      set_time_seri
60
61      # Add PV syst
62      SmartInverter
63
64      dss.text("solve")
65
66      # Read Energymeter results
67      energymeter_results = get_energymeter_results()
68      feeder_kwh_list.append(energymeter_results[0])
69      feeder_kvarh_list.append(energymeter_results[1])
70      loads_kwh_list.append(energymeter_results[2])
71      losses_kwh_list.append(energymeter_results[3])
72      pv_kwh_list.append(energymeter_results[4])
```

```
15  def set_baseline():
16      dss.text("New Energymeter.m1 Line.ln5815900-1 1")
17      dss.text("Set Maxiterations=20")
18      dss.text("set maxcontrolit=100")
19      dss.text("Batchedit Load..* daily=default")
```

Process (process.py)

```
55  for smart_inverter_function in smart_inverter_functions_list:
56      # Process for each smart inverter function
57      dss.text(f"Compile [{dss_file}]")
58      set_baseline()
59      set_time_series_simulation()
60
61      # Add PV system an
62      SmartInverterFunct                                    nction)
63
64      dss.text("solve")
65
66      # Read Energymeter results
67      energymeter_results = get_energymeter_results()
68      feeder_kwh_list.append(energymeter_results[0])
69      feeder_kvarh_list.append(energymeter_results[1])
70      loads_kwh_list.append(energymeter_results[2])
71      losses_kwh_list.append(energymeter_results[3])
72      pv_kwh_list.append(energymeter_results[4])
```

```
22  def set_time_series_simulation():
23      dss.text("set controlmode=Static")
24      dss.text("set mode=daily")
25      dss.text("set number=24")
26      dss.text("set stepsize=1h")
```

Process (process.py)

```python
for smart_inverter_function in smart_inverter_functions_list:
    # Process for each smart inverter function
    dss.text(f"Compile [{dss_file}]")
    set_baseline()
    set_time_series_simulation()

    # Add PV system and the smart inverter function
    SmartInverterFunction(dss, bus, 12.47, 8000, 8000, smart_inverter_function)

    dss.text("solve")

    # Read Energymeter results
    energymeter_results = get_ener
    feeder_kwh_list.append(energym
    feeder_kvarh_list.append(energ
    loads_kwh_list.append(energyme
    losses_kwh_list.append(energym
    pv_kwh_list.append(energymeter
```

```python
# -*- coding: utf-8 -*-
# @Time    : 8/9/2021 12:10 PM
# @Author  : Paulo Radatz
# @Email   : pradatz@epri.com
# @File    : smart_inverter_functions.py
# @Software: PyCharm


class SmartInverterFunction:

    def __init__(self, dss, bus, kV_base, kW_rated, kVA, smart_inverter_function):

        self.dss = dss
        self.bus = bus
        self.kV_base = kV_base
        self.kW_rated = kW_rated
        self.kVA = kVA
        self.smart_inverter_function = smart_inverter_function

        self.__define_3ph_pvsystem_with_transformer()

        self.__define_smart_inverter_function()
```

Process (process.py)

```python
for smart_inverter_function in smart_inverter_functions_list:
    # Process for each smart inverter function
    dss.text(f"Compile [{dss_file}]")
    set_baseline()
    set_time_series_simulation()

    # Add PV system and the smart inverter function
    SmartInverterFunction(dss, bus, 12.47, 8000, 8000, smart_inverter_function)

    dss.text("solve")

    # Read Energymeter results
    energymeter
    feeder_kwh
    feeder_kva
    loads_kwh_
    losses_kwh
    pv_kwh_lis
```

```python
def get_energymeter_results():
    dss.meters_write_name("m1")
    feeder_kwh = dss.meters_register_values()[0]
    feeder_kvarh = dss.meters_register_values()[1]
    loads_kwh = dss.meters_register_values()[4]
    losses_kwh = dss.meters_register_values()[12]
    pv_kwh = loads_kwh + losses_kwh - feeder_kwh

    return feeder_kwh, feeder_kvarh, loads_kwh, losses_kwh, pv_kwh
```

Process (process.py)

```python
for smart_inverter_function in smart_inverter_functions_list:
    # Process for each smart inverter function
    dss.text(f"Compile [{dss_file}]")
    set_baseline()
    set_time_series_simulation()

    # Add PV system and the smart inverter function
    SmartInverterFunction(dss, bus, 12.47, 8000, 8000, smart_inverter_function)

    dss.text("solve")

    # Read Energymeter results
    energymeter_results = get_energymeter_results()
    feeder_kwh_list.append(energymeter_results[0])
    feeder_kvarh_list.append(energymeter_results[1])
    loads_kwh_list.append(energymeter_results[2])
    losses_kwh_list.append(energymeter_results[3])
    pv_kwh_list.append(energymeter_results[4])
```

Process (process.py)

```
74  # Save results in a csv file
75  dict_to_df = dict()
76  dict_to_df["smart_inverter_function"] = smart_inverter_functions_list
77  dict_to_df["feeder_kwh"] = feeder_kwh_list
78  dict_to_df["feeder_kvarh"] = feeder_kvarh_list
79  dict_to_df["loads_kwh"] = loads_kwh_list
80  dict_to_df["losses_kwh"] = losses_kwh_list
81  dict_to_df["pv_kwh"] = pv_kwh_list
82
83  df = pd.DataFrame().from_dict(dict_to_df)
84
85  output_file = pathlib.Path(script_path).joinpath("outputs", "results.csv")
86  df.to_csv(output_file, index=False)
```

Output File

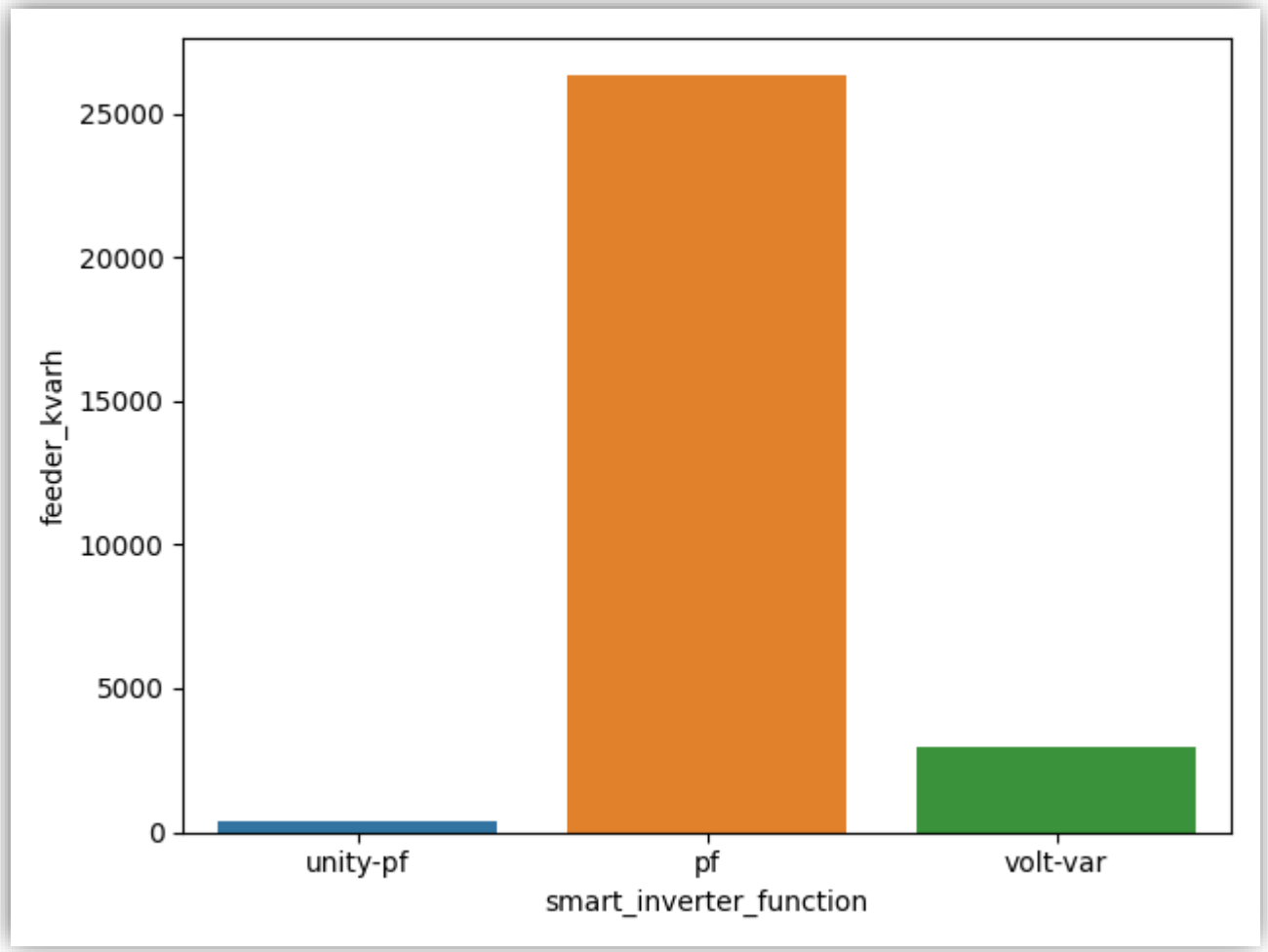
	A	B	C	D	E	F
1	smart_inverter_function	feeder_kwh	feeder_kvarh	loads_kwh	losses_kwh	pv_kwh
2	unity-pf	170313.7	355.2	217351.3	16082.4	63119.9
3	pf	173624.2	26311.8	217351.4	17072.7	60799.8
4	volt-var	170478.8	2982.7	217350.9	16171.1	63043.3
5						

"…\TimesSeries_OpenDSS\outputs\results.csv"

Data Analysis (data_analysis.py)

```python
# -*- coding: utf-8 -*-
# @Time    : 8/9/2021 1:32 PM
# @Author  : Paulo Radatz
# @Email   : pradatz@epri.com
# @File    : data_analysis.py
# @Software: PyCharm

import pandas as pd
import pathlib
import os
import matplotlib.pyplot as plt
import seaborn as sns

script_path = os.path.dirname(os.path.abspath(__file__))
output_file = pathlib.Path(script_path).joinpath("outputs", "results.csv")

df = pd.read_csv(output_file, index_col=0)

for column in df.columns:
    sns.barplot(x=df.index, y=column, data=df)
    plt.show()
```

Example of Results

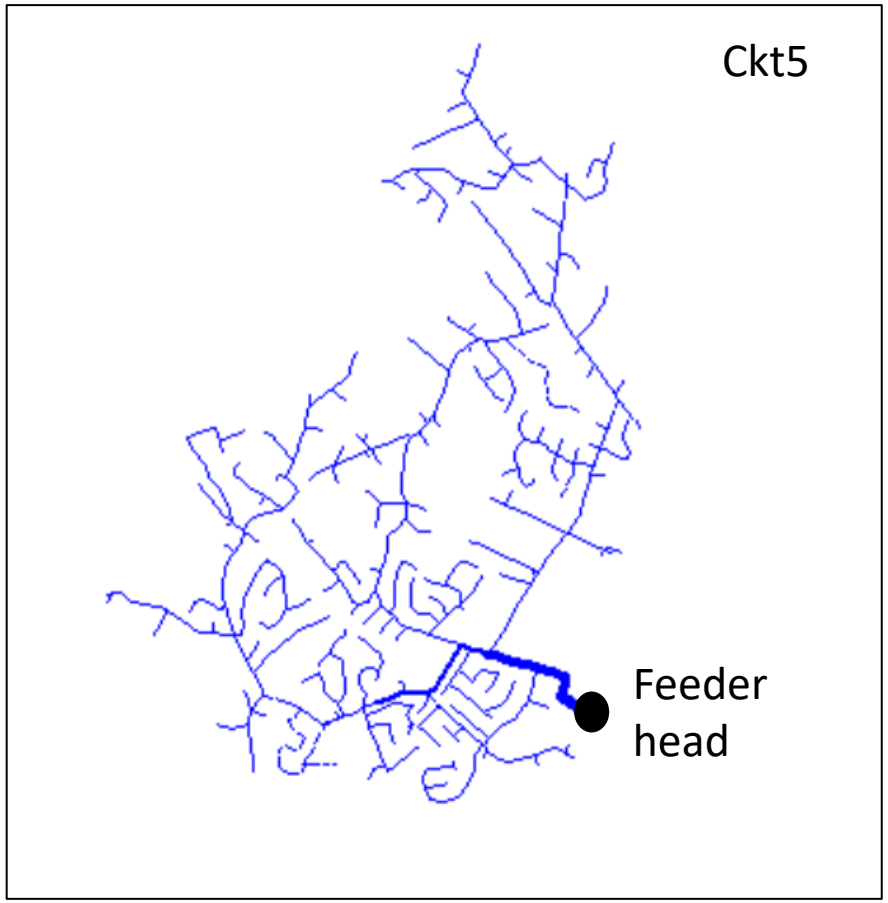


Example #2: Hosting Capacity Analysis

What is Hosting Capacity?

- Hosting capacity is defined as an estimate of the amount of DER that can be accommodated anywhere on the distribution system without adversely impacting power quality or reliability under existing control configurations and without requiring infrastructure upgrades.
  - The Hosting Capacity Process: https://www.epri.com/research/products/000000003002019750
  - EPRI's Distribution Resource Integration and Value Estimation (DRIVE) Tool : https://www.youtube.com/watch?v=n4papbQWp28&t=17s
  - Hosting Capacity Guidebook: 2020 Edition: https://www.epri.com/research/products/000000003002019000

Feeder



- Voltage regulating devices
  - Regulators are off
  - Capacitors are off
- Minimum load condition
  - 30% of max demand

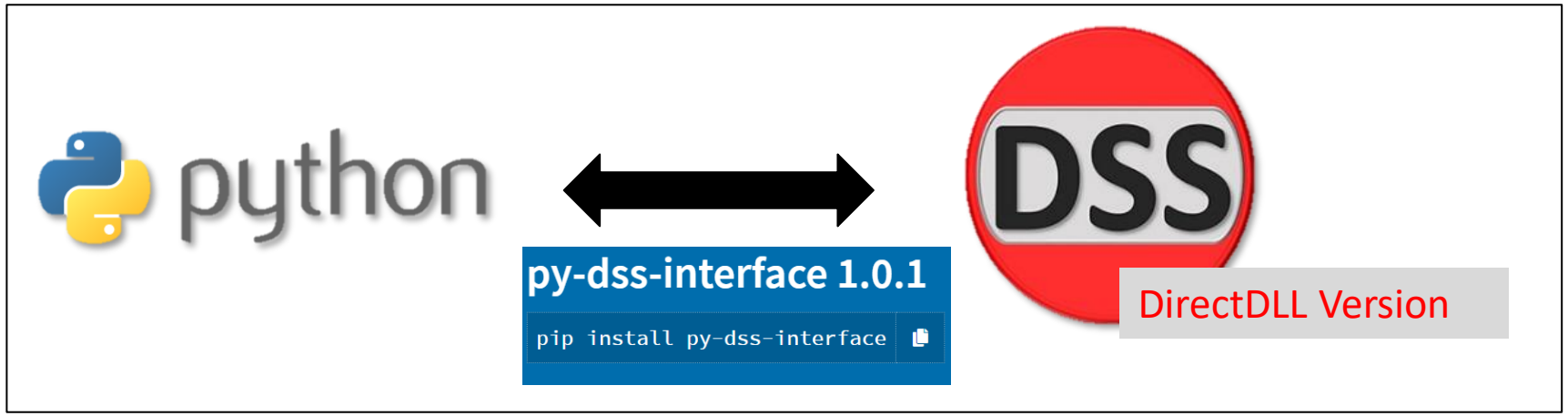
Ckt5 EPRI Test Circuit

Study Objectives

Provide the histogram of Overvoltage Centralized Hosting Capacity results for all three-phase medium voltage buses.

- Simulation
  - Perform the centralized hosting capacity process considering overvoltage as the only metric
  - Obtain OV HC for each 3-ph MV bus
- Data Analysis
  - Plot the histogram of the OV HC for all considered buses

Simulation Part



- Process
  1. Compile Circuit
  2. Get all 3-ph MV buses
  3. For each bus
     1. Compile circuit
     2. Set baseline
     3. Run OV HC process
     4. Save OV HC
  4. Save (Buses; OV HC results)

Process (process_hc.py)

```
40  script_path = os.path.dirname(os.path.abspath(__file__))
41  dss_file = pathlib.Path(script_path).joinpath("Feeders", "ckt5", "Master_ckt5.dss")
42
43  dss = py_dss_interface.DSSDLL()
44
45  dss.text(f"Compile [{dss_file}]")
46  mv_buses, mv_bus_voltage_dict = get_3ph_mv_buses()
```

Process (process_hc.py)

```
40  script_path = os.path.dirname(os.path.abspath(__file__))
41  dss_file = pathlib.Path(script_path).joinpath("Feeders", "ckt5", "Master_ckt5.dss")
42
43  dss = py_dss_interface.DSSDLL()
44
45  dss.text(f"Compile [{dss_file}]")
46  mv_buses, mv_bus_voltage_dict = get_3ph_mv_buses()
```

```
def get_3ph_mv_buses():
    mv_buses = list()
    mv_bus_voltage_dict = dict()

    buses = dss.circuit_all_bus_names()

    for bus in buses:
        dss.circuit_set_active_bus(bus)
        if bus == "sourcebus":
            pass
        elif dss.bus_kv_base() >= 1.0 and len(dss.bus_nodes()) == 3:
            mv_buses.append(bus)
            mv_bus_voltage_dict[bus] = dss.bus_kv_base() * math.sqrt(3)

    return mv_buses, mv_bus_voltage_dict
```

Process (process_hc.py)

```
48  bus_name_list = []
49  hc_list = []
50  for bus in mv_buses:
51      dss.text(f"Compile [{dss_file}]")
52      set_baseline()
53      bus_name_list.append(bus)
54      hc_obj = HostingCapacity(dss, bus, 0.3)
55      p_max = hc_obj.ov_criteria_hc_calc(mv_bus_voltage_dict[bus], 1.05, 100)
56      hc_list.append(p_max)
```

Process (process_hc.py)

```
48  bus_name_list = []
49  hc_list = []
50  for bus in mv_buses:
51      dss.text(f"Compile [{dss_file}]")
52      set_baseline()
53      bus_name_list.append(bus)
54      hc_obj = HostingCapacity(dss, bus, 0.3)
55      p_max = hc_obj.ov_criteria_hc_calc(mv_bus_volta
56      hc_list.append(p_max)
```

```
1   # -*- coding: utf-8 -*-
2   # @Time      : 8/6/2021 3:00 PM
3   # @Author    : Paulo Radatz
4   # @Email     : pradatz@epri.com
5   # @File      : Methods.py
6   # @Software: PyCharm
7
8
9   class HostingCapacity:
10
11      def __init__(self, dss, bus, load_mult):
12          self.dss = dss
13          self.bus = bus
14          self.load_mult = load_mult
```

Process (process_hc.py)

```
48  bus_name_list = []
49  hc_list = []
50  for bus in mv_buses:
51      dss.text(f"Compile [{dss_file}]")
52      set_baseline()
53      bus_name_list.append(bus)
54      hc_obj = HostingCapacity(dss, bus, 0.3)
55      p_max = hc_obj.ov_criteria_hc_calc(mv
56      hc_list.append(p_max)
```

```
    def ov_criteria_hc_calc(self, gen_kv, ov_threshold_pu, p_step, max_p=10000):
        self.__new_gen(gen_kv, p_step)

        i = 0
        ov_violation = False

        while not ov_violation and i * p_step <= max_p:
            i += 1
            self.__increment_generator_size(p_step * i)
            self.dss.text("solve")
            max_v = self.__get_max_feeder_voltage()

            if max_v > ov_threshold_pu:
                ov_violation = True

        if ov_violation:
            return (i - 1) * p_step
        else:
            return max_p
```

Process (process_hc.py)

```
48  bus_name_list = []
49  hc_list = []
50  for bus in mv_buses:
51      dss.text(f"Compile [{dss_file}]")
52      set_baseline()
53      bus_name_list.append(bus)
54      hc_obj = HostingCapacity(dss, bus, 0.3)
55      p_max = hc_obj.ov_criteria_hc_calc(mv_bus_voltage_dict[bus], 1.05, 100)
56      hc_list.append(p_max)
```

Process (process_hc.py)

```
58  # Save results in a csv file
59  dict_to_df = dict()
60  dict_to_df["Bus_name"] = bus_name_list
61  dict_to_df["Hosting_Capacity_kW"] = hc_list
62
63  df = pd.DataFrame().from_dict(dict_to_df)
64
65  output_file = pathlib.Path(script_path).joinpath("outputs", "results.csv")
66  df.to_csv(output_file, index=False)
```

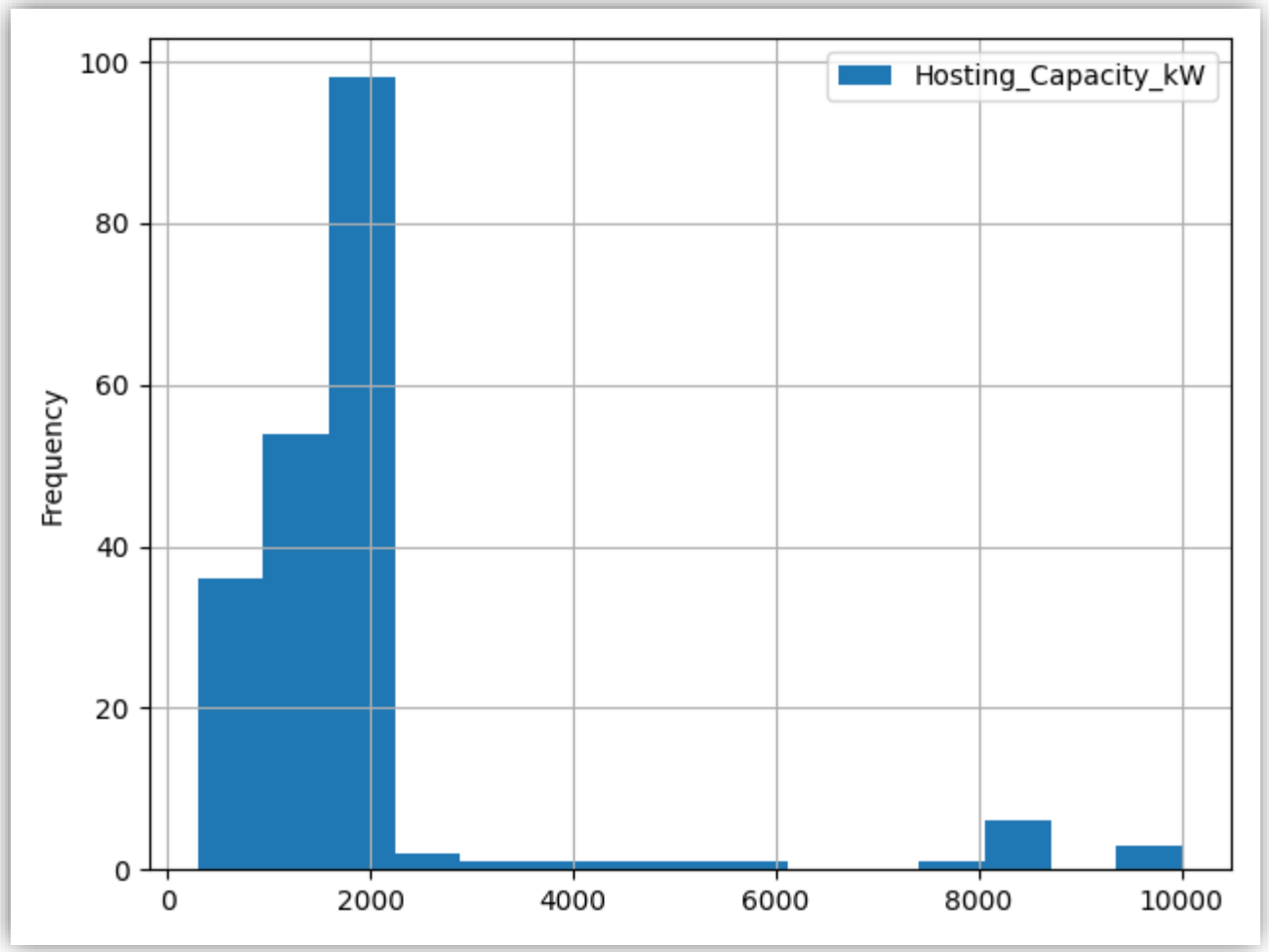
"…\HostingCapacity_OpenDSS\outputs\results.csv"

Data Analysis (data_analysis.py)

```python
# -*- coding: utf-8 -*-
# @Time    : 8/9/2021 11:07 AM
# @Author  : Paulo Radatz
# @Email   : pradatz@epri.com
# @File    : data_analysis.py
# @Software: PyCharm

import pandas as pd
import pathlib
import os
import matplotlib.pyplot as plt

script_path = os.path.dirname(os.path.abspath(__file__))
output_file = pathlib.Path(script_path).joinpath("outputs", "results.csv")

df = pd.read_csv(output_file, index_col=0)

print(df["Hosting_Capacity_kW"].describe())

df.plot.hist(bins=15)
plt.grid()
plt.show()
```

Data Analysis (data_analysis.py)

count	205.000000
mean	1841.951220
std	1725.954154
min	300.000000
25%	1300.000000
50%	1600.000000
75%	1700.000000
max	10000.000000



Closing Remarks

How is OpenDSS used ?

DER integration studies:

- Large-scale **PV systems** (centralized),
- Residential rooftop **PV systems** (distributed),
- **Wind** turbines,
- **Electric vehicles,**
- Customer-owned **storage,**
- Large-scale energy **storage**, etc.

- **Planning studies**:
  - Time-varying feeder behaviors,
  - Load modeling: (load allocations, load growth, AMI data, etc.),
  - System reliability study, etc.

- **Advanced System Controls**:
  - Distributed Energy Resource Management Systems (DERMS),
  - Volt-var optimization (VVO),
  - Conservation Voltage Reduction (CVR),
  - Fault Location, Isolation, and Service Restoration (FLISR), etc.

Many other applications of OpenDSS!!

2021 OpenDSS training instructors

Roger Dugan

Celso Rocha

Andres Ovalle

Paulo Radatz

Davis Montenegro

Jeremiah Deboever

Training recordings / material can be found on www.epri.com/opendss