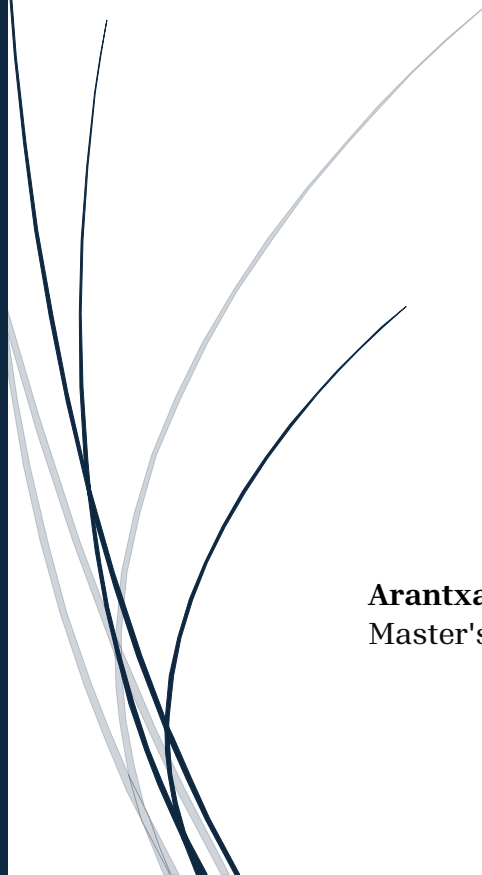


4/23/2025

# ECE 2774 – DC Power Flow Enhancement

Technical Manual for Python-based  
Power Flow Simulator



**Arantxa E. Better**, *Power Systems Engineer / P&C Engineer*  
Master's Candidate in Electrical and Computer Engineering  
Graduate Student Researcher  
Swanson School of Engineering  
University of Pittsburgh

## Table of Contents

### Contents

Table of Contents .....	1
1. Purpose and Theoretical Background .....	2
1.1. Feature Overview, Justification & References .....	2
Key Features: .....	2
Importance: .....	2
References: .....	2
1.2. Implementation Details .....	3
1.2.1. New Class: DCPowerFlowSolver .....	3
1.2.2. Circuit Class Enhancements .....	3
1.2.3. Solver Integration in the user interface (Seven Bus System) .....	4
2. Testing and Validation Instructions .....	4
2.1. Test Case 1: 3-Bus System (Manual Verification) .....	4
Manual Calculation: .....	4
Python simulator result: .....	5
2.2. Test Case 2 – 7-Bus System Validation (PowerWorld Comparison) .....	6
PowerWold: .....	6
Python simulator result: .....	6
2.3. Conclusion of Testing and Validation .....	7

# 1. Purpose and Theoretical Background

The objective of this enhancement is to implement a DC Power Flow solver into an existing Python-based power system simulator. DC Power Flow is a simplified linear approximation of AC Power Flow, widely used in transmission-level studies due to its computational efficiency and clarity in analyzing active power distribution.

This model is based on the following assumptions:

- Flat voltage magnitude:  $|V| = 1.0$
- Small angle differences:  $\sin(\delta_i - \delta_j) \approx \delta_i - \delta_j$
- Negligible resistance:  $R \rightarrow 0$  X contributes
- Only real power P is considered (no Q, no voltage magnitudes)

The governing equation becomes:

$$\delta = -B'^{-1} \cdot P$$

Where:

- B' is the reduced susceptance matrix (removes slack bus row/column)
- $\delta$  is the bus voltage angle vector (radians or degrees)
- P is the net power injection vector (in per-unit)

## 1.1.Feature Overview, Justification & References

Key Features:

- Dynamic generation of B' using Ybus imaginary components
- Per-unit conversion of power injections
- Slack bus elimination with reinsertion of reference angle (0°)
- Real-time printout of matrices and angle results
- Angle output in both radians and degrees for validation

Importance:

DC Power Flow is a core tool in grid operation and planning:

- Used in contingency screening
- Initial guess for AC power flow
- Faster than Newton-Raphson methods

References:

- PowerWorld DC Power Flow Documentation

- J. D. Glover, T. J. Overbye, and M. S. Sarma, Power System Analysis and Design, 6th ed., SI Edition. Boston, MA, USA: Cengage Learning, 2017.

## 1.2.Implementation Details

To implement DC Power Flow capability within the object-oriented simulator, a new class and several utility functions were introduced.

### 1.2.1. New Class: DCPowerFlowSolver

A new class named DCPowerFlowSolver was added. This class performs the linearized DC power flow using the following logic:

1. Retrieve Ybus from the circuit and extract its imaginary part to construct the susceptance matrix BBB
2. Reduce B by removing the slack bus row and column to get B'B'B'
3. Build the power injection vector PPP using bus generation and load data in p.u.
4. Solve the equation  $\delta = -B'^{-1} \cdot P$  for unknown bus voltage angles
5. Re-insert the slack bus angle (0°) and return full angle profile in radians and degrees

The class includes optional debug printouts to display:

- Susceptance matrix B'
- Net real power injection vector P
- Voltage angles in degrees

### 1.2.2. Circuit Class Enhancements

To support the construction of P, the following two methods were added to the Circuit class:

```
def get_bus_generation(self, bus_name):
    total_gen = 0.0
    for gen in self.generators.values():
        if gen.bus.name == bus_name:
            total_gen += gen.real_power
    return total_gen

def get_bus_load(self, bus_name):
    total_load = 0.0
    for load in self.loads.values():
        if load.bus.name == bus_name:
            total_load += load.real_power
    return total_load
```

These return the aggregated real power generated or consumed at a given bus, enabling automated power vector construction within the solver.

### 1.2.3. Solver Integration in the user interface (Seven Bus System)

To invoke the DC power flow analysis on any system built through the simulator, the following three lines are used:

```
from DCPowerFlowSolver import DCPowerFlowSolver

# Run DC Power Flow
dc_solver = DCPowerFlowSolver(circuit)
dc_solver.solve()
dc_solver.display_results()
```

This allows the user to run power flow immediately after the network is built, using any topology or combination of buses, lines, generators, and loads.

## 2. Testing and Validation Instructions

### 2.1. Test Case 1: 3-Bus System (Manual Verification)

This test case was designed to validate the DC Power Flow implementation using a minimal 3-bus configuration that allows for easy manual calculations, see Figure 1.

The system consists of:

- A slack bus (Bus 1) at 20 kV with a generator grounded in wye configuration,
- An intermediate bus (Bus 2) connected to the slack through a transformer,
- A load bus (Bus 3) with a 110 MW load,
- A transmission line connecting Bus 2 to Bus 3.

All components were modeled using base values of 100 MVA for power and appropriate per-unit conversions were applied.

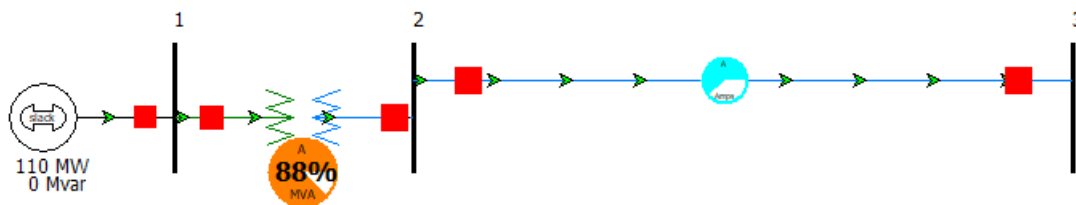


Figure 1. Three Bus System

Manual Calculation:

Starting from the Ybus:

Name	Bus 1	Bus 2	Bus 3
1	1.46 - j14.63	-1.46 + j14.63	
2	-1.46 + j14.63	11.84 - j46.72	-10.38 + j32.14
3		-10.38 + j32.14	10.38 - j32.09

Reduced reactance matrix:

$$B' = \begin{bmatrix} -46.72 & 31.14 \\ 32.14 & -32.09 \end{bmatrix}$$

Inverted reactance matrix:

$$B'^{-1} = \begin{bmatrix} -0.0688 & -0.0689 \\ -0.0689 & -0.1002 \end{bmatrix}$$

Real Power vector in p.u.:

$$P = \begin{bmatrix} 0 \\ -1.1 \end{bmatrix}$$

Applying the linearized equation of the DC power flow method:

$$\delta = -B'^{-1} \cdot P = - \begin{bmatrix} -0.0688 & -0.0689 \\ -0.0689 & -0.1002 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ -1.1 \end{bmatrix} = \begin{bmatrix} -4.3444 \\ -6.3152 \end{bmatrix}$$

Python simulator result:

```

--- Ybus (for Power Flow Analysis) ---
          Bus 1          Bus 2          Bus 3
Bus 1  1.46329-14.63290j  -1.463290+14.632900j   0.000000+ 0.000000j
Bus 2  -1.46329+14.63290j  11.838769-46.724378j  -10.375479+32.137885j
Bus 3   0.00000+ 0.00000j  -10.375479+32.137885j   10.375479-32.091478j
>> Entered DCPowerFlowSolver.solve()

--- Susceptance Matrix B' (p.u.^-1) ---
          Bus 2    Bus 3
Bus 2  -46.7244   32.1379
Bus 3   32.1379  -32.0915

--- Net Real Power Injection Vector P (p.u.) ---
Bus 2: 0.0000
Bus 3: -1.1000

--- DC Power Flow Results (Voltage Angles in Degrees) ---
Bus 1: 0.0000°
Bus 2: -4.3409°
Bus 3: -6.3111°

```

These values match the manual solution with a difference of less than 0.005°, confirming the numerical accuracy of the solver.

## 2.2. Test Case 2 – 7-Bus System Validation (PowerWorld Comparison)

This test verifies the accuracy of the DC Power Flow enhancement by comparing simulation results against PowerWorld for the same 7-bus system used in Project 2.

The 7-bus system previously modeled in Project 2 was reused for this test. It includes:

- One **slack bus** (Bus 1),
- One **PV generator bus**,
- Five **PQ buses** with distributed loads,
- Multiple **transformers** and **transmission lines** modeled with geometry, impedance, and grounding properties.

All base quantities (e.g., 100 MVA system base) and component data remained unchanged.

PowerWorld:

Number	Name	Area Name	Nom kV	PU Volt	Volt (kV)	Angle (Deg)	Load MW	Load Mvar	Gen MW	Gen Mvar	Switched Shunts Mvar	Act G Shunt MW	Act B Shunt Mvar	Area Num	Zone Num
1	1	1	20.00	1.00000	20.000	0.00			115.87	85.81		0.00	0.00	1	1
2	2	1	230.00	0.93692	215.491	-4.44						0.00	0.00	1	1
3	3	1	230.00	0.92049	211.712	-5.46	110.00	50.00				0.00	0.00	1	1
4	4	1	230.00	0.92980	213.853	-4.70	100.00	70.00				0.00	0.00	1	1
5	5	1	230.00	0.92672	213.147	-4.83	100.00	65.00				0.00	0.00	1	1
6	6	1	230.00	0.93968	216.126	-3.95						0.00	0.00	1	1
7	7	1	18.00	0.99999	18.000	2.15	0.00	0.00	200.00	108.79		0.00	0.00	1	1

Python simulator result:

```

--- Ybus (for Power Flow Analysis) ---
      Bus 1      Bus 2      Bus 3      Bus 4      Bus 5      Bus 6      Bus 7
Bus 1  1.46329-14.63290j  -1.46329+ 14.63290j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j
Bus 2  -1.46329+14.63290j  37.777466-127.050527j  -10.375479+32.137885j  -25.938697+ 80.344712j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j
Bus 3  0.00000+ 0.00000j  -10.375479+ 32.137885j  23.344827-72.226709j  0.00000+ 0.00000j  -12.969349+ 40.172356j  0.00000+ 0.00000j  0.00000+ 0.00000j
Bus 4  0.00000+ 0.00000j  -25.938697+ 80.344712j  0.00000+ 0.00000j  46.319102-143.352043j  -7.411056+ 22.955632j  -12.969349+ 40.172356j  0.00000+ 0.00000j
Bus 5  0.00000+ 0.00000j  0.00000+ 0.00000j  -12.969349+40.172356j  -7.411056+ 22.955632j  46.319102-143.352043j  -25.938697+ 80.344712j  0.00000+ 0.00000j
Bus 6  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  -12.969349+ 40.172356j  -25.938697+ 80.344712j  40.489864-139.443204j
Bus 7  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  0.00000+ 0.00000j  -1.581819+ 18.981824j  1.581819-18.981824j

>> Entered DCPowerFlowSolver.solve()

--- Susceptance Matrix B' (p.u.^-1) ---
      Bus 2      Bus 3      Bus 4      Bus 5      Bus 6      Bus 7
Bus 2  -127.0505  32.1379  80.3447  0.0000  0.0000  0.0000
Bus 3   32.1379 -72.2267  0.0000  40.1724  0.0000  0.0000
Bus 4   80.3447  0.0000 -143.3520  22.9556  40.1724  0.0000
Bus 5   0.0000  40.1724  22.9556 -143.3520  80.3447  0.0000
Bus 6   0.0000  0.0000  40.1724  80.3447 -139.4432  18.9818
Bus 7   0.0000  0.0000  0.0000  0.0000  18.9818 -18.9818

--- Net Real Power Injection Vector P (p.u.) ---
Bus 2: 0.0000
Bus 3: -1.1000
Bus 4: -1.0000
Bus 5: -1.0000
Bus 6: 0.0000
Bus 7: 2.0000

--- DC Power Flow Results (Voltage Angles in Degrees) ---
Bus 1: 0.0000°
Bus 2: -4.4544°
Bus 3: -5.6124°
Bus 4: -4.7989°
Bus 5: -4.9582°
Bus 6: -3.9561°
Bus 7: 2.0808°

```

### 2.3. Conclusion of Testing and Validation

The DC Power Flow enhancement was tested using both a manually solvable 3-bus system and a more complex 7-bus system previously developed in Project 2. The 3-bus system results were manually verified using matrix-based calculations, while the 7-bus system outputs were compared against PowerWorld's DC power flow results.

Both test cases confirmed excellent agreement, with maximum observed angle deviations below  $0.01^\circ$ , affirming the numerical stability and correctness of the implementation. The results demonstrate that the enhancement integrates seamlessly with the existing simulator architecture and can scale to realistic power system models.

The full source code for this project, including test cases and the DC Power Flow solver implementation, is available at:

[https://github.com/Kamisama-D/Main\\_Simulator/tree/Ara\\_Project\\_3](https://github.com/Kamisama-D/Main_Simulator/tree/Ara_Project_3)