Introduction to the power flow analysis

ELEC0447 - Analysis of electric power and energy systems

Bertrand Cornélusse

October 3, 2025

Overview

1. Introduction
2. The power flow equations
3. Power flow solution method

Introduction

What is a power flow analysis? I

Power flow (or load flow) analysis aims at determining the *electrical state of an electrical power system*, when information about power generated or consumed is available at nodes of the network, and considering that the voltage level is regulated at some buses.

This type of analysis is commonly used by power companies for planning and operation purposes.

- If voltage magnitude and angles were measured at all buses,
  - then it would boil down to solving a set of simple linear equations.
- In a similar way, mesh or nodal analysis could be used if we had a full model of the system,

What is a power flow analysis? II

- even without all voltage measurements.
- But here the situation is different, because we mainly have access to *power* measurements.
  - The system is no more linear.

Power flow problem statement I

Determine **the voltage at every bus**, assuming we have a power system composed of transmission lines connecting the following bus types:

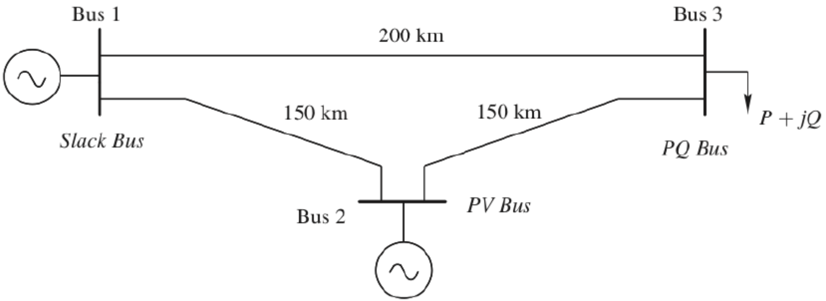
- *PQ buses* are typically loads where active and reactive power are measured
  - it can also be generation where voltage is not regulated (e.g. renewable generation)
- *PV buses* where the active power and the voltage are specified
  - these are typically generators
- one *slack bus* that sets the reference for the voltage magnitudes and angles (it is usually at 1 pu)
  - P and Q can take any value to reach the power balance in the system.

Power flow problem statement II

Branch currents and losses can be determined from the voltages (magnitudes and phases).

Note: as we will see, PV buses must be swithed to PQ buses in case they reach a limit of their capability curve.

A first tiny example I



A first tiny example II

Buses:

- Bus 1 is the slack, with V = 1 pu
- Bus 2 is a PV bus, with V regulated at 1.05 pu and drawing P= 2 pu
- Bus 3 is a PQ bus, consumes P = 5 pu and Q = 1 pu.

Lines:

- X = 0.376 Ohm/km (at 60 Hz)
- R = 0.037 Ohm/km
- Shunt susceptances are ignored (4.5e-6 S/km)

Voltage base (3-phase): 345 kV, Power base (3-phase): 100 MVA

Modeling the system in PandaPower

We will see how to describe mathematically the problem and how to solve it in the next sections.

Usually, however, engineers use existing modeling and solution software. One of those, which is open source and accessible through Python scripts, is PandaPower [1].

Link to the PandaPower model

Creation of the power flow model with PandaPower I

```
# Import PandaPower package and name it pp
import pandapower as pp

# Create an empty network object
net = pp.create_empty_network()

# Per unit bases (3-phase values)
Pbase = 100  # MVA
Vbase = 345 # kV

# Create buses (geodata are coordinates for graphical representation)
b1 = pp.create_bus(net, vn_kv=Vbase, name="Bus 1", geodata=(0,1))
b2 = pp.create_bus(net, vn_kv=Vbase, name="Bus 2", geodata=(2.5,0))
b3 = pp.create_bus(net, vn_kv=Vbase, name="Bus 3", geodata=(5,1))
```

Creation of the power flow model with PandaPower II

```
# Create bus elements
pp.create_ext_grid(net, bus=b1, vm_pu=1.00, name="Grid Connection")
pp.create_load(net, bus=b3, p_mw=5*Pbase, q_mvar=1*Pbase, name="Load")
pp.create_gen(net, bus=b2, p_mw=2*Pbase, vm_pu=1.05, name="PV")

# Create branch elements.
# Here I neglect shunct capacitances.
Zbase = Vbase**2 / Pbase
X_km = 0.376
R_km = 0.037
l12_km = 150
l23_km = 150
l31_km = 200

```

Creation of the power flow model with PandaPower III

```
pp.create_line_from_parameters(net, name="line1", from_bus = b1, to_bus = b2,
        length_km=112_km, r_ohm_per_km = R_km, x_ohm_per_km = X_km,
        c_nf_per_km = 0, max_i_ka = 0.2)
pp.create_line_from_parameters(net, name="line2", from_bus = b2, to_bus = b3,
        length_km=123_km, r_ohm_per_km = R_km, x_ohm_per_km = X_km,
        c_nf_per_km = 0, max_i_ka = 0.2)
pp.create_line_from_parameters(net, name="line3", from_bus = b3, to_bus = b1,
        length_km=131_km, r_ohm_per_km = R_km, x_ohm_per_km = X_km,
        c_nf_per_km = 0, max_i_ka = 0.2)

# Solve the model
pp.runpp(net)
```

Result of the tiny example using pandapower

	vm_pu	va_degree	p_mw	q_mvar
0	1.00	0.00	-308.38	81.61
1	1.05	-2.07	-200.00	-266.74
2	0.98	-8.79	500.00	100.00

Are there losses?

Results for the lines:

	p_from_mw	q_from_mvar	p_to_mw	q_to_mvar	pl_mw	ql_mvar	i_from_ka	i_to_ka
0	68.99	-110.87	-68.20	118.95	0.80	8.08	0.22	0.22
1	268.20	147.79	-264.23	-107.49	3.97	40.30	0.49	0.49
2	-235.77	7.49	239.38	29.26	3.62	36.75	0.40	0.40

The power flow equations

The power flow equations I

- Let $\mathcal{N}$ be the set of buses of the network
- Some buses are interconnected by transmission lines, given by their π models
- Let Y_{kG} be the sum of admittances connected between node k and the ground:
  - the shunt admittances of the lines incident to k, and the admittances of the devices connected at node k if any.
- For two nodes k and m, let Z_{km} be the series impedance of the line connecting them and $Y_{km} = Z_{km}^{-1}$ ($Y_{km} = 0$ if there is no line)

The power flow equations II

The current injection at node k is

$$\bar{I}_k = Y_{kG}\bar{V}_k + \sum_{m\in\mathcal{N}\setminus k} (\bar{V}_k - \bar{V}_m)Y_{km} \tag{1}$$

The power flow equations III

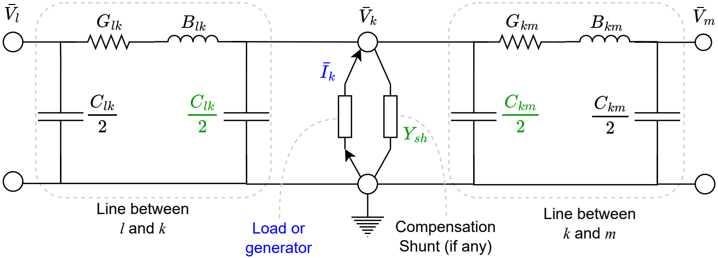


Figure 1: Illustration of current injection at bus k as a function of exchanges with neighbor buses (l and m) and shunt compensation device. Devices and lines connected to buses l and m are not shown.

The power flow equations IV

This last equation can be rewritten as

$$\bar{I}_k = \left(Y_{kG} + \sum_{m \in \mathcal{N} \setminus k} Y_{km} \right) \bar{V}_k - \sum_{m \in \mathcal{N} \setminus k} Y_{km} \bar{V}_m$$

which highlights the possibility to write in matrix form

$$\bar{\mathbf{I}} = \mathbf{Y} \bar{\mathbf{V}} \tag{2}$$

with $\bar{\mathbf{I}}$ and $\bar{\mathbf{V}}$ the vectors of bus current injections and bus voltages, respectively.

The power flow equations V

The *admittance matrix* $\mathbf{Y}$ can be determined by inspection:

- Element y_{kk} is the sum of the admittances incident to bus k
- Element $y_{km}, m \neq k$, is the opposite of the sum of the admittances connecting bus k to bus m.

However, remember that we have power measurements only (and voltage magnitudes at a few PV buses). So we can derive

$$\mathbf{P} + j\mathbf{Q} = \bar{\mathbf{V}} \circ \bar{\mathbf{I}}^{\star} = \bar{\mathbf{V}} \circ \mathbf{Y}^{\star}\bar{\mathbf{V}}^{\star} \tag{3}$$

The power flow equations VI

where $\mathbf{P}$ and $\mathbf{Q}$ are the vectors of active and reactive power injections, respectively, and $\circ$ denotes the elementwise product.

If we develop this relation for a node k, we have:

$$P_k = G_{kk}V_k^2 + V_k \sum_{m \in \mathcal{N} \setminus k} V_m (G_{km} \cos\theta_{km} + B_{km} \sin\theta_{km}) = p_k(\bar{\mathbf{V}}) \tag{4}$$

$$Q_k = -B_{kk}V_k^2 + V_k \sum_{m \in \mathcal{N} \setminus k} V_m (G_{km} \sin\theta_{km} - B_{km} \cos\theta_{km}) = q_k(\bar{\mathbf{V}}) \tag{5}$$

with

The power flow equations VII

- $Y_{km} = G_{km} + jB_{km}$
- $Y_{kk} = G_{kk} + jB_{kk}$ is the sum of the admittances from bus k to ground
- $\theta_{km} = \theta_k - \theta_m$ the phase difference between voltages at nodes k and m

Number of equations and unknowns I

If there are n buses in total, among which n_{PQ} PQ buses, n_{PV} PV buses and one slack bus, hence

$$n = n_{PQ} + n_{PV} + 1,$$

then

- $\mathbf{P}$ is known for $n_{PQ} + n_{PV}$ buses (all but the slack)
- Elements of $\mathbf{Q}$ are known for the n_{PQ} PQ buses
- Voltage magnitude is known at PV buses and at the slack bus
- Voltage angle is known at the slack bus.

Number of equations and unknowns II

In total, there are $2n$ equations for $2n$ unknowns:

- $n - 1$ voltage angles,
- n_{PQ} voltage magnitudes,
- $n_{PV} + 1$ reactive powers,
- 1 active power.

Power flow solution method

Power flow solution method

Let

- $\mathbf{P}^0$ be the active powers specified at the $\mathcal{N}_{PQ} \cup \mathcal{N}_{PV}$ buses
- $\mathbf{Q}^0$ be the reactive powers specified at the $\mathcal{N}_{PQ}$ buses.

To find $\bar{\mathbf{V}}$, we must solve

$$\begin{aligned} P_k^0 - p_k(\bar{\mathbf{V}}) = 0,\ \forall k \in \mathcal{N}_{PQ} \cup \mathcal{N}_{PV} \\ Q_k^0 - q_k(\bar{\mathbf{V}}) = 0,\ \forall k \in \mathcal{N}_{PQ} \end{aligned}$$

which is a set of $2n_{PQ} + n_{PV}$ non-linear equations.

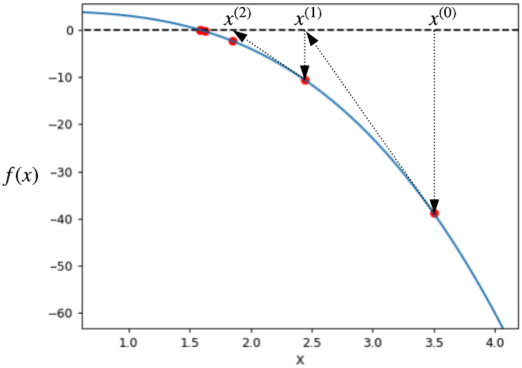
The most widespread method to solve this system is the *Newton-Raphson method*.

Newton-Raphson example in 1D I

- Let's assume we want to solve $c - f(x) = 0$ with f a non-linear function.
- We start with a first guess for x, $x^{(0)}$, at iteration $i = 0$
- Then, while $|c - f(x^{(i)})| > \epsilon$:
  - $x^{(i+1)} = x^{(i)} + \frac{c - f(x^{(i)})}{f'(x^{(i)})}$
  - $i \leftarrow i + 1$

Newton-Raphson example in 1D II

For $c = 4$ and $f(x) = x^3$:



The *convergence is quadratic* if we start with x(0) "close" to the solution.

Link to Python implementation

Newton-Raphson for the power flow problem I

We apply exactly the same idea to our problem, except that we are in dimension $2n_{PQ} + n_{PV}$.

Hence we must compute partial derivatives to compute the update steps:

$$\bar{\mathbf{V}}_x^{(i+1)} = \bar{\mathbf{V}}_x^{(i)} + \underbrace{\left[\mathbf{J}(\bar{\mathbf{V}}^{(i)})\right]^{-1}(\mathbf{F}^0 - \mathbf{f}(\bar{\mathbf{V}}^{(i)}))}_{\Delta\bar{\mathbf{V}}_x}$$

where

- $\mathbf{F}^0$ gathers the measured active powers at buses in $\mathcal{N}_{PQ} \cup \mathcal{N}_{PV}$ and reactive powers at buses $\mathcal{N}_{PQ}$

Newton-Raphson for the power flow problem II

- $\mathbf{f}(\bar{\mathbf{V}})$ gathers the active and power flow equations at the corresponding buses
- $\bar{\mathbf{V}}_x$ is the subvector of $\bar{\mathbf{V}}$ that gathers the unknwon voltage magnitudes and angles at the the corresponding buses
- $\mathbf{J}(\bar{\mathbf{V}})$ is the jacobian of $\mathbf{f}$, of size $(2n_{PQ} + n_{PV}) \times (2n_{PQ} + n_{PV})$

Remarks

- In practice, instead of computing the inverse of the Jacobian, we solve the system

$$\mathbf{J}(\bar{\mathbf{V}}^{(i)})\Delta\bar{\mathbf{V}}_x = \mathbf{F}^0 - \mathbf{f}(\bar{\mathbf{V}}^{(i)}) \tag{6}$$

  to get the update step
- The Jacobian is often sparse, since a bus is connected to a few neighbors; it is very important to take into account the sparsity properties in practical implementations
- The Jacobian is not necessarily updated at every iteration, especially close to convergence

Illustration I

Let's reconsider the 3-bus example we already modeled with PandaPower.

We will

1. Compute the line impedances of the system $Z_{1,2}, Z_{1,3}, Z_{2,3}$ where $Z_{i,j}$ is the impedance of the line between bus-i and bus-j.
2. Calculate the base impedance Z_B and compute the per-unit line impedances.
3. Compose the per-unit admittance matrix of the system $\mathbf{Y}_{pu} = \mathbf{G}_{pu} + j\mathbf{B}_{pu}$.
4. Initialize the unknown voltages of the Newton-Raphson procedure:
   - there are two unknowns at the PQ bus (V_3, θ_3) and one unknown at the PV bus (θ_2).

Illustration II

- initialize the unknown magnitudes at 1 pu and unknown phases at 0 rad (first arbitrary estimation).

5. Perform one iteration in the Newton-Raphson procedure.
   - Use the active and reactive power equations (4) and (5) at the nodes where active and reactive power are known, i.e. estimate $p_2\left(\bar{V}^{(k)}\right), p_3\left(\bar{V}^{(k)}\right)$ and $q_3\left(\bar{V}^{(k)}\right)$, for $k=0$
   - Compute the **error** of the initial step: subtract estimates from known values.
   - Compute the Jacobian matrix $\mathbf{J}$ of the system and evaluate it at $\overline{\mathbf{V}}^{(0)}$.
   - Find the **update step** $\Delta\overline{\mathbf{V}}$ by solving (6).
   - Update the estimated voltages magnitudes and phases with the **update step**
6. Repeat the process until the **error** is low enough.

Illustration III

Exercise

- Write down the expression of $p_2\left(\bar{V}^{(k)}\right)$
- What are the dimensions of the Jacobian?
- Write down the expression of 3 elements of the Jacobian.
- Understand and run the Python notebook **WARNING: indices start at 0 in Python, so bus 1 is at index 0, etc.**
- Compare the results to the PandaPower model.

Fast decoupled power flow I

Remember that

- active power flow is mostly a function of voltage angles
- reactive power flow is mostly a function of voltage magnitudes

If we apply these ideas stricly, we can subdivide the problem in two much simpler subproblems:

- one problem for angles, based on the active power measurements and the sub-Jacobian containing the partial derivatives of the active power flow equations w.r.t. angles

Fast decoupled power flow II

- one problem for magnitudes, based on the reactive power measurements and the sub-Jacobian containing the partial derivatives of the reactive power flow equations w.r.t. magnitudes

This procedure, through the sub-Jacobian that are computed, also provide information useful for *sensitivity analysis*.

DC power flow I

"Direct Current" power flow is a further simplification:

- it is assumed that the impact of the reactance of lines is much bigger than the impact of their resistance, and shunt conductances are neglected
- voltage magnitudes are assumed equal to $1pu$
- angle differences are small
- active power losses are neglected, reactive power flows as well

DC power flow II

$$P_k = \sum_{m \in \mathcal{N} \setminus k} B_{km} \theta_{km}$$

for every bus but the slack bus, which sets the angle difference, and collects the algebraic sum of all other injected powers.

In matrix form, with $\mathbf{Y}$ the admittance matrix defined before:

$$\mathbf{P} = \Im(\mathbf{Y})\theta$$

This is usefull for fast simulations, or when including a power flow model in an optimization problem, e.g. day-ahead market coupling.

Illustration of DC power flow

Application to the 3-bus case

- Show how to obtain the relaxation of the DC power flow
- Apply the DC approximation to the 3 bus case
- Understand the code and analyze the results: Python notebook

References I

L. Thurner, A. Scheidler, F. Schäfer, J.-H. Menke, J. Dollichon, F. Meier, S. Meinecke, and M. Braun, "pandapower—an open-source python tool for convenient modeling, analysis, and optimization of electric power systems," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6510–6521, 2018.