Steady-state validation

iPST/PowSyBl day, 2018-05-25

Nicolas Omont
nicolas.omont@rte-france.com

Why a steady-state validation?

➔ **Internal:**

- PowSyBl is a platform integrating many load-flows and optimal power flows (Hades, Helm LF, Eurostag LF, AMPL-based OPFs, PyPSA)
- How to ensure the consistency of results across all tools? How to validate the good integration of a network computation module?
- Numerical consistency required to chain the computations (Initialization of dynamic simulations from load flow results…)
  - The needed precision is much higher than the precision of physical measures.

➔ **External:**

- Network states are exchanged between TSOs. How to ensure the consistency of network simulations done by TSOs on the same network data?
- Current processes in ENTSO-E OPDE (Operational Planning Data environment) focus on the conformity of data to the CGMES standard, not on the consistency of the SSH (Steady State Hypothesis) profile content.
- A validation is needed to ensure models used to interpret CGMES data in each European TSO tools are identical.
  - Differences may have significant consequences (Cross-border capacity reduction…)

How to validate steady-state results?

→ Thanks to engineering practice:

- For more than 50 years, engineers have modelled and computed power systems steady-state.
- They agree on most of what is a “good” steady-state results.

→ It is possible to derive rules from this practice

- Strict rules when a common understanding exist.
- Lenient ones if no consensus exist (for example, in case of inconsistent input data).
- In the “grey” zone in-between, try to propose an explicit rule.
  - PowSyBl is open-source: engineers can refer to the details of the rule to discuss it / re-implement it in their own tool.

Why not compare the steady states directly?

➔ A single discrepancy has impact on all results

- As all results change, it is difficult to spot the discrepancy.
- And it is impossible to solve all discrepancies (some leniency is needed).

➔ There are several consistent steady-state for the same input data:

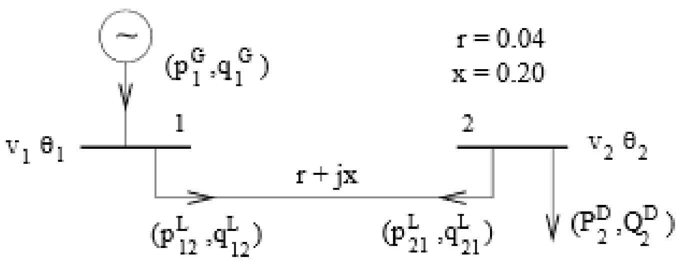


Fig. 1. Two Bus System

Fixed load: P_2^D = 352.5 MW
Q_2^D =-358.0 MVAr

TWO OPF SOLUTIONS FOR THE 2 BUS PROBLEM

	Bus	v (p.u.)	θ (deg)	p^G (MW)	q^G (MVAr)
S_1	1	0.950	0.00	456.55	162.25
	2	0.985	−65.01		
S_2	1	0.951	0.00	444.08	99.91
	2	1.050	−58.12		

Low voltage, high losses steady-state

High voltage, low losses steady state

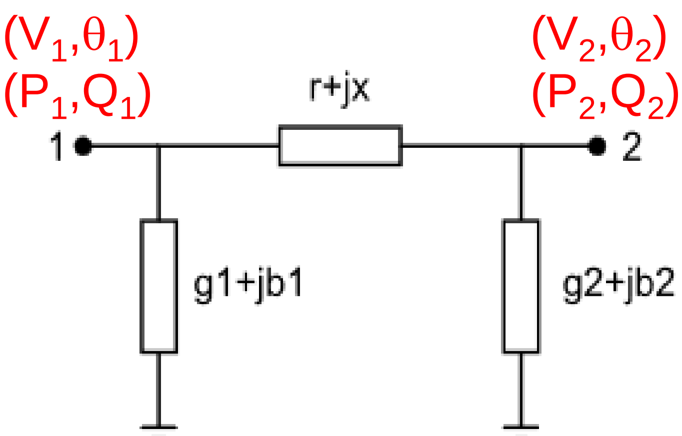
(Extracted from “Local Solutions of Optimal Power Flow”, Bukhsh at al.)

“Line” validation example


```
public interface Line
extends Branch<Line>, LineCharasteristics<Line>
```

An AC line.

The equivalent π model used is:



To create a line, see `LineAdder`

PowSyBl line javadoc

$$(P_{1est},Q_{1est})=f\big((V_1,\theta_1),(V_2,\theta_2),(r,j,g1,b1,g2,b2)\big)$$

$$(P_{2est},Q_{2est})=f\big((V_1,\theta_1),(V_2,\theta_2),(r,j,g1,b1,g2,b2)\big)$$

(Equations in the open-source code)

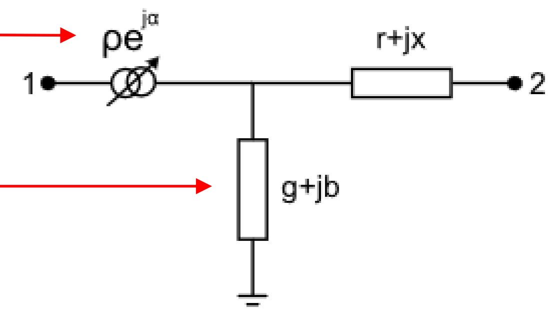
$$(P_{1est},Q_{1est})=(P1,Q1)$$
$$(P_{2est},Q_{2est})=(P2,Q2)$$

Other validated elements

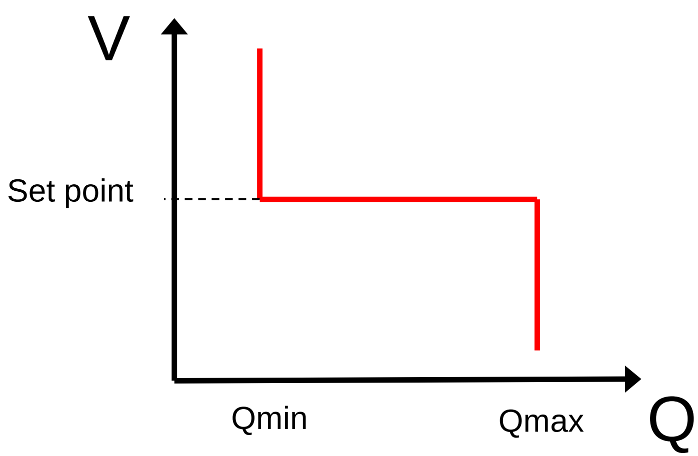
- **2w transformers:**

Ideal transformer on side 1

Shunt impedance on side 1 only



  - 3w transformers: to be done
- **Regulating tap-changer transformers**
  - Based on dead-band estimation
    - Assumption: control of lower voltage radial network.
    - Details in the documentation.
- **Generation units (and SVCs): PV/PQ model**

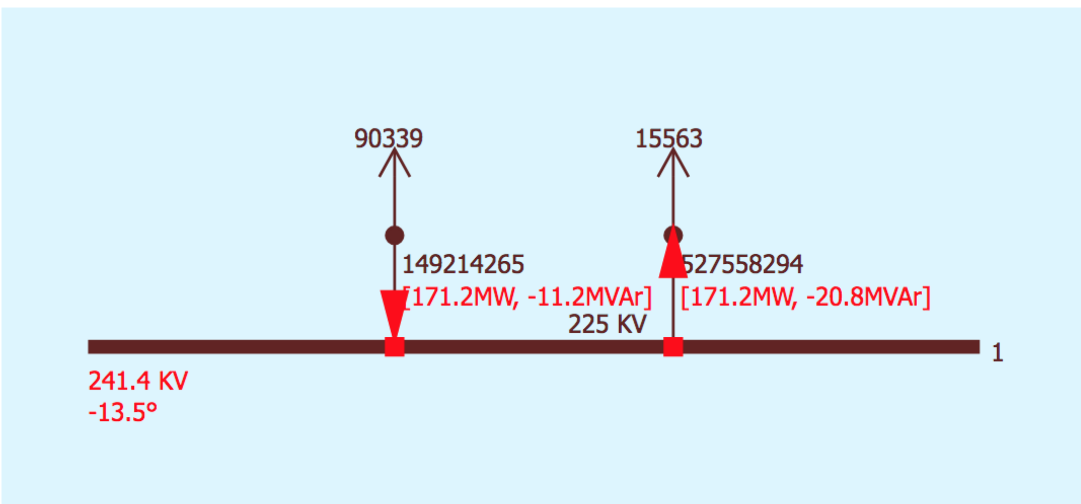


Steady-state ⟺ On the red line

Completion feature and energy conservation

➔ **Main load-flow tools only provide voltage and angles (flows are not mandatory in CGMES)**

- Using the validation equations, the platform provide a way to recompute the flows.
- Nodal active and reactive energy balance can then be checked.



-11.2 MVAr ≠ -20.8 MVAr !

Example (1/2)

```
itesla@machine:~/nicoomon$ itools loadflow-validation --case-file recollement-auto-
20180214-0830-enrichi.xiidm --output-folder ~/val --output-folder CSV --verbose
Loading case recollement-auto-20180214-0830-enrichi.xiidm
Running pre-loadflow validation on network recollement-auto-20180214-0830-enrichi
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: SHUNTS - result: fail
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: SVCS - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: GENERATORS - result: fail
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: BUSES - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: FLOWS - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: TWTS - result: success
Running loadflow on network recollement-auto-20180214-0830-enrichi
Running post-loadflow validation on network recollement-auto-20180214-0830-enrichi
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: SHUNTS - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: SVCS - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: GENERATORS - result: fail
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: BUSES - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: FLOWS - result: success
Validate load-flow results of network recollement-auto-20180214-0830-enrichi -
validation type: TWTS - result: success
```

Validation of steady-state results stored in file.

Issue with generators and shunts

Validation of recomputed results with Hades

Issue with generators only

Example (2/2)

recollement-auto-20180214-0830-enrichi GENERATORS check

Stored steady-state | Computed S.S.

	id	p	q	v	targetP	targetQ	targetV	connected	voltageRegulatorOn	minQ	maxQ	mainComponent	validation	p_postLF	q_postLF	v_postLF	targetP _postLF	targetQ_postLF	targetV _postLF	connected_postLF	voltageRegulatorOn_postLF	minQ_postLF	maxQ_postLF	mainComponent_postLF	validation_postLF
408	Gen 1	-7	-0	64	-7	-0	64	true	true	-0,1	0,1	true	succ	-7	-0	64	-7	-0	64	true	true	-0,1	0,1	true	success
409	Gen 2	0	inv	inv	0	0	0	false	false	-2,84	2,84	true	succ	inv	inv	inv	0	0	0	false	false	-2,84	2,84	true	success
410	**Gen 3**	**0**	**-9**	**65**	**0**	**-9**	**65**	**true**	**true**	**-5,8**	**5,8**	**true**	**fail**	**0**	**-9**	**65**	**0**	**-9**	**65**	**true**	**true**	**-5,8**	**5,8**	**true**	**fail**
411	Gen 4	0	0	65	0	0	65	true	false	0	0	true	succ	0	0	65	0	0	65	true	false	0	0	true	success
412	Gen 5	0,18	12	65	0,18	12	65	true	true	-9,009	18,56	true	succ	0,18	12	65	0,18	12	65	true	true	-9,009	18,56	true	success
413	Gen 6	0	inv	inv	0	0	0	false	false	-8,3	18,7	true	succ	inv	inv	inv	0	0	0	false	false	-8,3	18,7	true	success

Voltage v (65 kV) is equal to voltage set point targetV (65 kV) but reactive power q (-9 MVAr) is out of [minQ,maxQ] range ([-5.8,5.8]).

The issue was relatively easy to spot.

(But not so easy to solve… The unit is used as a synchronous compensator (P=0), thus below the minimum power, therefore it is impossible to compute reactive power bounds, which are functions of P and thus defined only if P is in [minP,maxP]…).

Conclusion

→ **A steady-state validation tool has two roles:**

- Consistency of PowSyBl computation modules.
- Analyse and validation of steady states provides by other tools.

→ **We think it can be used to improve the consistency of data exchanged by european TSO.**