LF ENERGY POWER GRID MODEL

Power Grid Model DS

Empowering the Energy Transition through Fast and Flexible Network Simulation

LF ENERGY

Introduction

We firmly believe smart data science solutions are essential in enabling the energy transition

Introduction

Thijs Baaijen

Python Software Engineer

Jaap Schouten

Product Owner Data Science

LF ENERGY

alliander

🚀 Announcement

Today we are launching Power Grid Model DS

New exciting package extending the Power Grid Model Suite to allow for broader Data Science integration

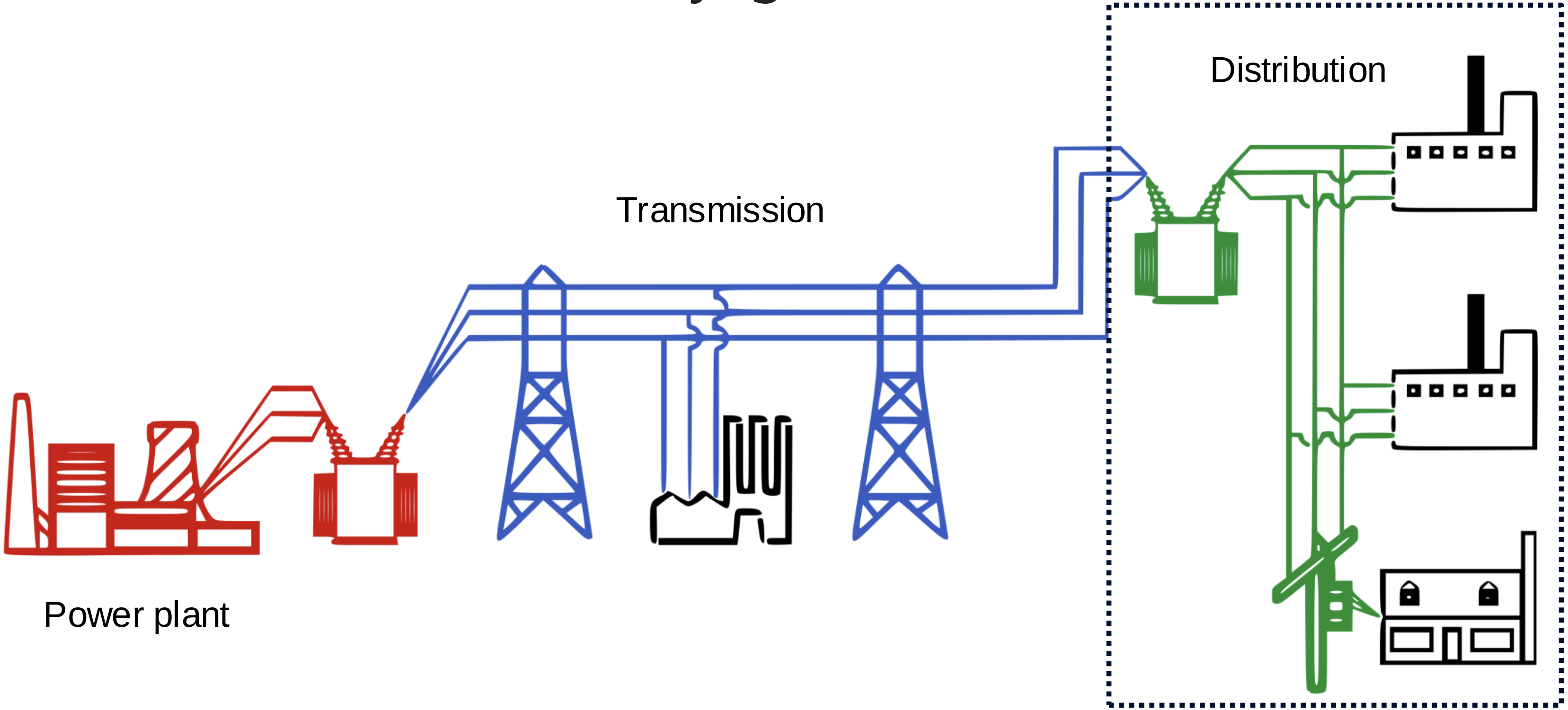
Power Grid Model

Power Grid Model is a high-performance Python/C++ library for steady-state distribution power system analysis, supporting state estimation, power flow, and short circuit calculations. With applications in grid planning, expansion, reliability, and congestion studies, Power Grid Model is an open-source initiative under LF Energy, driving innovation in energy.

How is the electricity network loaded? Where do we see contingencies coming?

LFENERGY

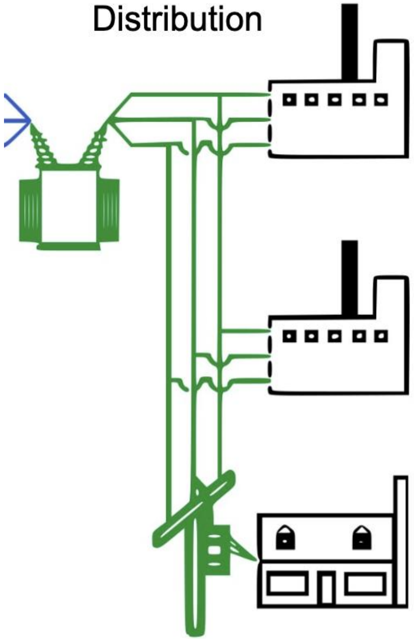
What is an electricity grid



https://commons.wikimedia.org/w/index.php?curid=5577847

LFENERGY

Nodes, Lines & Transformers



https://www.alliander.com

Busbar/switchgear units

↓

Nodes

https://www.alliander.com

Medium voltage cables

↓

Lines

...and **Transformers**

Power Grid Model - Data Science Toolkit

We have a fast load flow, let’s solve problems with it!

Within our teams at Alliander we are continuously using these calculations to

Determine the impact on the network **tomorrow**

And this needs lots of simulations ..., we do millions of power flows

Power Grid Model - Data Science Toolkit

We have a fast load flow, let’s solve problems with it!

Within our teams at Alliander we are continuously using these calculations to

Plan **maintenance** and rerouting in case of **outages**

And this needs lots of simulations …, we do millions of power flows

Power Grid Model - Data Science Toolkit

We have a fast load flow, let's solve problems with it!

Within our teams at Alliander we are continuously using these calculations to

Determine the impact of **future customers** in the next 10 years

And this needs lots of simulations ..., we do millions of power flows

Power Grid Model - Data Science Toolkit

We have a fast load flow, let's solve problems with it!

Within our teams at Alliander we are continuously using these calculations to

Make **strategic** decisions, for the next 40 years

And this needs lots of simulations ..., we do millions of power flows

Power Grid Model – In & Output

```python
# node
node = initialize_array(DatasetType.input, ComponentType.node, 3)
node["id"] = np.array([1, 2, 6])
node["u_rated"] = [10.5e3, 10.5e3, 10.5e3]

# line
line = initialize_array(DatasetType.input, ComponentType.line, 3)
line["id"] = [3, 5, 8]
line["from_node"] = [1, 2, 1]
line["to_node"] = [2, 6, 6]
line["from_status"] = [1, 1, 1]
line["to_status"] = [1, 1, 1]
line["r1"] = [0.25, 0.25, 0.25]
line["x1"] = [0.2, 0.2, 0.2]
line["c1"] = [10e-6, 10e-6, 10e-6]
line["tan1"] = [0.0, 0.0, 0.0]
line["i_n"] = [1000, 1000, 1000]

# load
sym_load = initialize_array(DatasetType.input, ComponentType.sym_load, 2)
sym_load["id"] = [4, 7]
sym_load["node"] = [2, 6]
sym_load["status"] = [1, 1]
sym_load["type"] = [LoadGenType.const_power, LoadGenType.const_power]
sym_load["p_specified"] = [20e6, 10e6]
sym_load["q_specified"] = [5e6, 2e6]

# source
source = initialize_array(DatasetType.input, ComponentType.source, 1)
source["id"] = [10]
source["node"] = [1]
source["status"] = [1]
source["u_ref"] = [1.0]

# all
input_data = {
    ComponentType.node: node,
    ComponentType.line: line,
    ComponentType.sym_load: sym_load,
    ComponentType.source: source,
}
```

```
------node result------
   id  energized      u_pu             u   u_angle             p             q
0   1          1  0.998988  10489.375043 -0.003039  3.121451e+07  6.991358e+06
1   2          1  0.952126   9997.325181 -0.026031 -2.000000e+07 -5.000000e+06
2   6          1  0.962096  10102.012975 -0.021895 -1.000000e+07 -2.000000e+06
------line result------
   id  energized   loading        p_from        q_from      i_from  \
0   3          1  0.985666  1.736010e+07  4.072097e+06  981.460041
1   5          1  0.205940 -3.365614e+06 -1.178649e+06  205.939917
2   8          1  0.783206  1.385441e+07  2.919262e+06  779.311446

         s_from          p_to          q_to        i_to          s_to
0  1.783129e+07 -1.663439e+07 -3.821351e+06  985.666324  1.706768e+07
1  3.566030e+06  3.396558e+06  8.861080e+05  200.617323  3.510241e+06
2  1.415863e+07 -1.339656e+07 -2.886108e+06  783.206396  1.370392e+07
```

Power Grid Model - Data Science Toolkit

- Are there any cycles in the network?
- What is the path from A to B?
- Which nodes are connected to the same feeder?

- What lines are overloaded?
- What voltage bounds are broken?

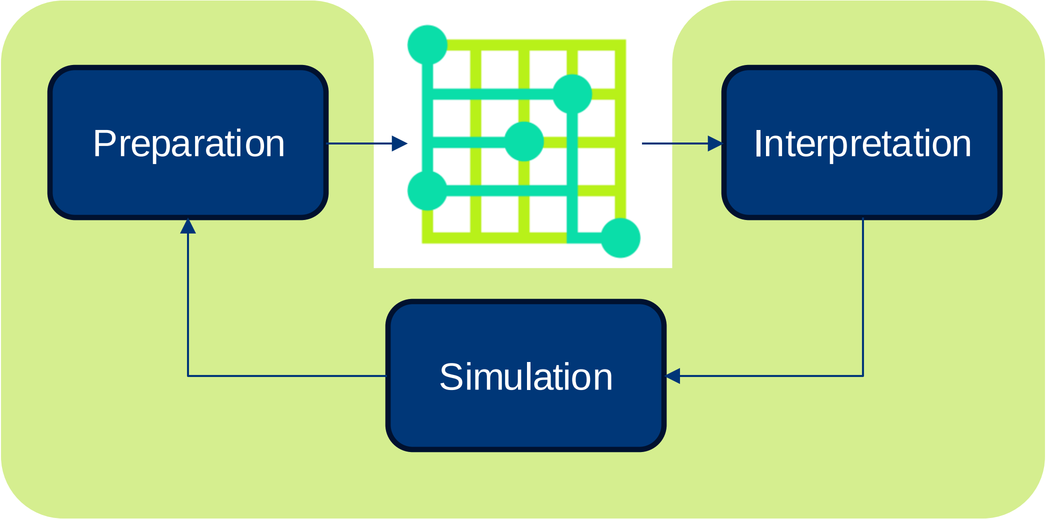
- How do we simulate changes on the network?

Power Grid Model - Data Science Toolkit

How do we use PGM-DS to do this?
Wrapper for PGM

1. Preparation
2. Interpretation
3. Simulation

💯 Ease of use



LF ENERGY

Power Grid Model - Data Science Toolkit

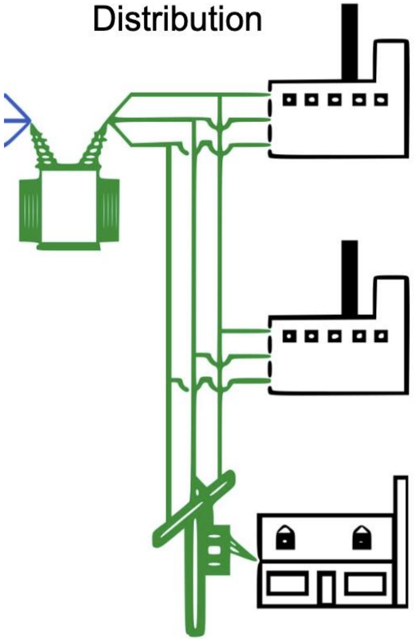
PGM-DS: **P**ower **G**rid **M**odel **D**ata **S**cience (Toolkit)

It’s a lightweight and fast Python wrapper around the Power Grid Model loadflow core:

Dependencies:

- Power Grid Model
- Numpy
- RustworkX

Nodes, Lines & Transformers



https://www.alliander.com

Busbar/switchgear units

↓

Nodes

https://www.alliander.com

Medium voltage cables

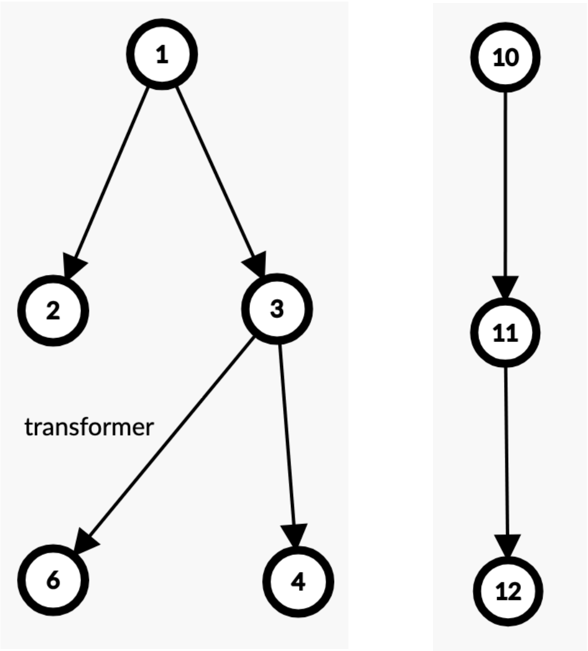
↓

Lines

...and **Transformers**

LFENERGY

Nodes, Lines & Transformers



https://csacademy.com/app/graph_editor/

NodeArray

id	u_rated	...
1	...	...
2	...	...
3	...	...
4	...	...
6	...	...
10	...	...
11	...	...
12	...	...

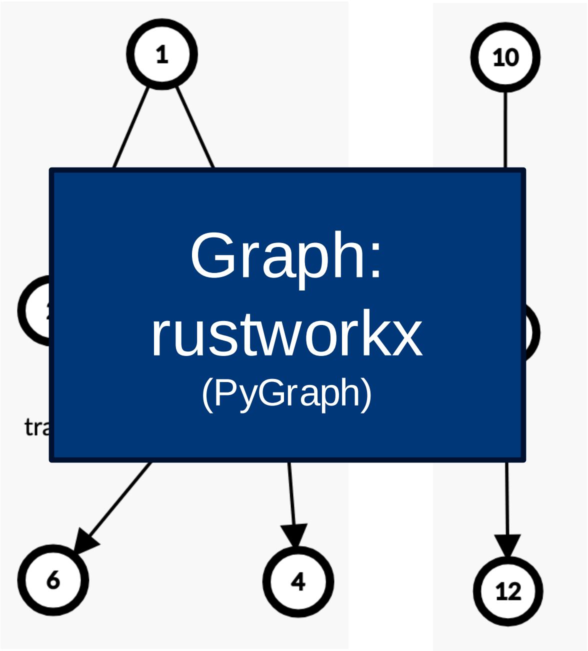
LineArray

id	from_node	to_node	from_status	to_status	r1	i_n	...
13	1	2	...	...	...	...	...
14	1	3	...	...	...	...	...
15	3	4	...	...	...	...	...
17	10	11	...	...	...	...	...
18	11	12	...	...	...	...	...

TransformerArray

id	from_node	to_node	from_status	to_status	u1	u2	sn	...
16	3	6	...	...	...	...	...	...

Two representations



Graph:
rustworkx
(PyGraph)

https://csacademy.com/app/graph_editor/

```
NodeArray
 id | u_rated | ...
 1  |   ...   | ...
 2  |   ...   | ...
 3  |   ...   | ...
 4  |   ...   | .
 6  |   ...   | .
 10 |   ...   | .
 11 |   ...   | .
 12 |   ...   | .

LineArray
 id | from_node |                                   i_n | ...
 13 |     1     |                                   ... | ...
 14 |     1     |                                   ... | ...
 15 |     3     |    4    |    ...    |    ...    | ... | ... | ...
 17 |    10     |   11    |    ...    |    ...    | ... | ... | ...
 18 |    11     |   12    |    ...    |    ...    | ... | ... | ...

TransformerArray
 id | from_node | to_node | from_status | to_status |  u1 |  u2 |  sn | ...
 16 |     3     |    6    |     ...     |    ...    | ... | ... | ... | ...
```

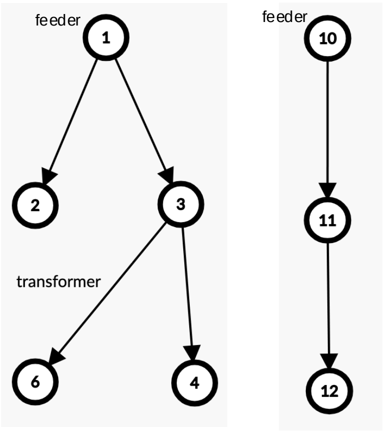
Arrays:
numpy
(structured arrays)

Graph questions

- Are there any cycles in the network?
- What is the path from A to B?
- Which nodes are connected to the same feeder?

Graph questions

- Are there any cycles in the network?
- What is the path from A to B?
- Which nodes are connected to the same feeder?




```python
class BaseGraphModel(ABC):
    def get_shortest_path(self, start_node_id: int, end_node_id: int) -> tuple[list[int], int]:
        """Calculate the shortest path between two nodes"""
    def get_all_paths(self, start_node_id: int, end_node_id: int) -> list[list[int]]:
        """Retrieves all paths between two nodes."""
    def get_connected(self, node_id: int) -> list[int]:
        """Find all nodes connected to the node_id"""
    def get_components(self) -> list[list[int]]:
        """Returns all separate components of the graph"""
    def get_downstream_nodes(self, node_id: int) -> list[int]:
        """Find all nodes below the node_id"""
    def find_fundamental_cycles(self) -> list[list[int]]:
        """Find all fundamental cycles in the graph."""
```

Array questions

power-grid-model

- What is the voltage at node A?
- Power flow analysis
- State Estimation

```python
from power_grid_model_ds import PowerGridModelInterface

core_interface = PowerGridModelInterface(grid=grid)

core_interface.create_input_from_grid()
core_interface.calculate_power_flow()
core_interface.update_grid()
```

```
LineArray
 id | from_node | to_node | from_status | to_status | r1 | x1 | c1 | tan1 | i_n
 13 |     1     |    2    |      1      |     1     | ... | ... | ... | ... | ...

TransformerArray
 id | from_node | to_node | from_status | to_status | u1 | u2 | sn | tap_size | uk | pk | i0 | p0 | winding_from | winding_to | clock | tap_side | tap_pos | tap_min | tap_max | tap_nom
 16 |     3     |    6    |      1      |     1     | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ...
```

Array questions

- Filtering
- Updating values
- Data type (dtype) inheritance
- Default values

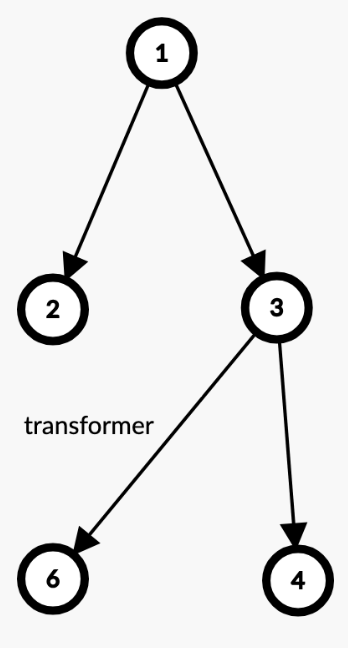
```python
# OOP-based approach with inheritance, defaults and other features
class MyCustomLineArray(LineArray):
    length_m: NDArray[np.float16]
    n_cores: NDArray[np.int8]
    material: NDArray[np.str_]
    is_healthy: NDArray[np.bool_]

    _defaults = {'length_m': 0, 'n_cores': 1}

# Filtering: numpy structured array
copper_mask = lines["material"] == 'copper'
healthy_mask = lines["is_healthy"]
healthy_copper_lines = lines[copper_mask & healthy_mask]

# Filtering: FancyArray
healthy_copper_lines = lines.filter(is_healthy=True, material='copper')
```

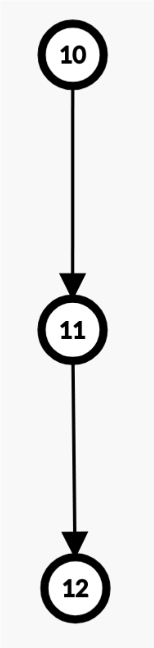
```python
class MyCustomArray(FancyArray):
    first_name: NDArray[np.str_]
    country: NDArray[np.str_]
    age: NDArray[np.int8]
```

```python
# Other methods
non_copper_lines = lines.exclude(material='copper')
lines.update_by_id([1, 2, 3], is_healthy=False)
```

Note: it's just a wrapper. Under the hood it's still a numpy structured array

Managing two representations





https://csacademy.com/app/graph_editor/

NodeArray

id	u_rated	...
1	...	...
2	...	...
3	...	...
4	...	...
6	...	...
10	...	...
11	...	...
12	...	...

LineArray

id	from_node	to_node	from_status	to_status	r1	i_n	...
13	1	2	...	...	...	...	...
14	1	3	...	...	...	...	...
15	3	4	...	...	...	...	...
17	10	11	...	...	...	...	...
18	11	12	...	...	...	...	...

TransformerArray

id	from_node	to_node	from_status	to_status	u1	u2	sn	...
16	3	6	...	...	...	...	...	...

Managing two representations

```python
@dataclass
class Grid:
    """Manages the Graph and Array representations of the power grid"""
    graphs: Graphs

    node: NodeArray
    line: LineArray
    transformer: TransformerArray

    # other power-grid-model arrays
    three_winding_transformer: ThreeWindingTransformerArray
    link: LinkArray
    source: SourceArray
    sym_load: SymLoadArray
    sym_gen: SymGenArray
    transformer_tap_regulator: TransformerTapRegulatorArray
    sym_power_sensor: SymPowerSensorArray
    sym_voltage_sensor: SymVoltageSensorArray
    asym_voltage_sensor: AsymVoltageSensorArray
```

```python
@dataclass
class Grid:
    """Manages the Graph and Array representations of the power grid"""
    ...
    def add_node(self, node: NodeArray) -> None:
        """Add a node to the grid"""
    def delete_node(self, node: NodeArray) -> None:
        """Delete a node from the grid"""
    def add_branch(self, branch: BranchArray) -> None:
        """Add a line/transformer/link to the grid"""
    def delete_branch(self, node: NodeArray) -> None:
        """Delete a line/transformer/link from the grid"""
    def activate(self, branch: BranchArray) -> None:
        """Power on a line/transformer/link (updates 'to_status')"""
    def deactivate(self, branch: BranchArray) -> None:
        """Power off a line/transformer/link (updates 'to_status')"""
```

LFENERGY

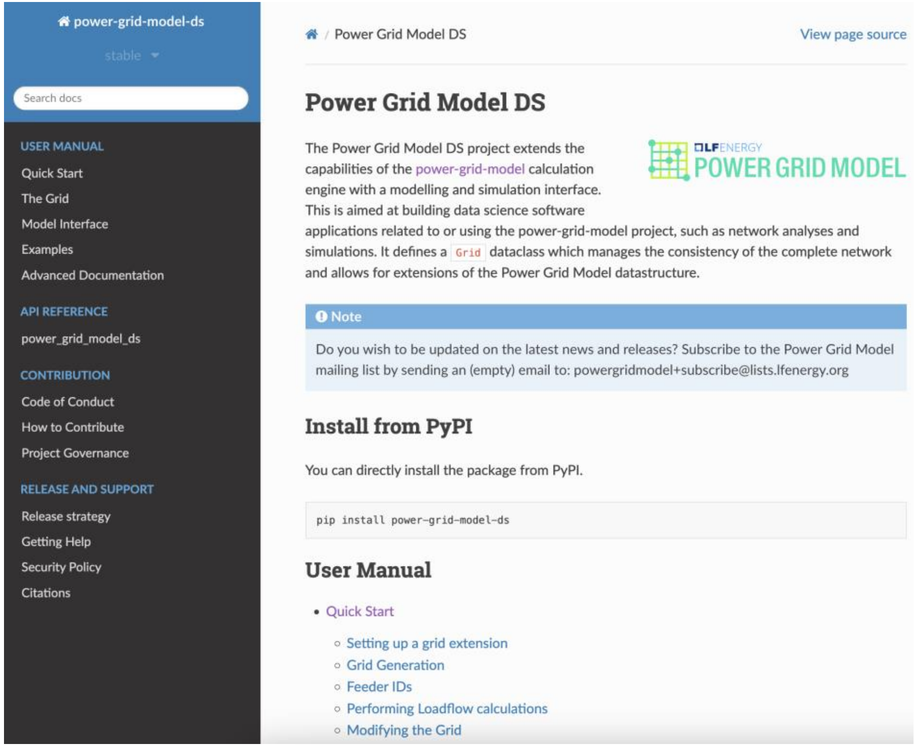
Wrap Up

As developers at Alliander we are excited to share these extensions with the community.

They have helped us a lot in building smart solutions for the energy transition, and hope it will strengthen collaboration such that we can build even stronger software with you, as an open source collaboration.

LFENERGY

Power Grid Model Data Science Toolkit



To stay up to date:

- Subscribe to our mailing list
- Attend the yearly Power Grid Model meetup
- ⭐

LF ENERGY