Considerations and Design Goals for Unbalanced Optimal Power Flow Benchmarks

Frederik Geth
GridQube
Brisbane, Australia

Archie C. Chapman
The University of Queensland
Brisbane, Australia

Rahmat Heidari
Energy Business Unit, CSIRO
Newcastle, Australia

Jordan Clark
GridQube
Brisbane, Australia

***Abstract*—Distribution network constrained mathematical optimization is key technology to enable advanced distribution network management. In recent years, a growing amount of articles on the development of such models and algorithms have been published. Benchmarking of different approaches is crucial to establish performance trade-offs between accuracy, reliability and computational intensity. Today, practitioners tend to take ad-hoc approaches, building on power flow data sets but adding customer extensions to establish information such as voltage/current/power bounds and to parameterize pre-defined objective functions. To foster progress in this field, in this work we discuss i) a number of design trade-offs and pitfalls related to benchmarking, ii) develop a data model and mathematical specification for (up-to) four-wire optimal power flow, and iii) develop some initial data sets. The data sets are provided through open-access initiatives under a creative commons license, and a reference implementation of the mathematical model is made available with a permissive license. For ease-of-use, and to maximize uptake in the community, we establish a tiered approach to the benchmark development with a multi-year plan.**

I. INTRODUCTION

Unbalanced Optimal Power Flow (UBOPF) is the framework underlying "distribution network physics"-constrained optimization models [1], [2]. In UBOPF, the steady-state ac multiconductor version of Kirchhoff's circuits laws is represented, thereby capturing electrical phenomena such as phase unbalance and neutral voltage shift. Phase unbalance necessitates the representation of impedance *matrices* (instead of scalars) for lines, phase connectivity between power delivery elements, and phase connectivity of loads and generators.

A. The need for benchmarking

Continual advances in computing hardware, algorithms that exploit this and the development of software environments for physics-based optimisation models now make UBOPF methods practically feasible at useful scales. Simultaneously, networks are hosting increasing amounts of *distributed energy resources*, such as rooftop PV and electric vehicles, which is growing the need for UBOPF in distribution network planning and operations. *Benchmarking* of UBOPF methods enables the scientific method in a variety of ways, leading to better-informed research strategies and more reliable dissemination of reproducible lessons learnt. For instance, it helps to:

- find accidental errors in implementations [3];
- profile algorithms in terms of optimality, scalability and occurrence of specific numerical issues;
- establish best-practice formulations;
- establish comparisons for heuristics or artificial intelligence/machine learning (AI/ML) models [1].

It is generally understood that validating and benchmarking new methods for power network simulation and optimization is crucial to understand their usefulness and impact. However, in this paper, we adopt the inverse perspective and focus on the development, and systematic organisation, of benchmark test cases. We focus on the UBOPF problem, but our approach is more general and considers a range of problems in unbalanced power networks. We note that data quality influences the representativeness of such models [4] as well as the performance of solver algorithms, so here pay particular attention to the data comprising benchmarks, including hidden and irreversible transformations, forced decompositions, and unsuitable approximations. Specifically, we consider models up to four-wire, but note that true n-wire models are available today in mature simulation engines such as OPENDSS.

Our aim is to further the development of datasets for the benchmarking of UBOPF, with a focus on defining appropriate *data models* that standardize ways to store asset engineering data, as well as datasets abiding these rules. Our aim is guided by three key observations of the state of the art. First, despite progress in the development of 3/4-wire unbalanced OPF engines [5], [6], benchmarking is still hard, due to (i) a lack of an accepted engineering data model, and, (ii) a lack of open data sets. As a workaround, authors frequently start from the IEEE test feeders and perform ad-hoc data extensions, including but not limited to voltage/power/current bounds and generator cost functions.

Second, it is noted that a mathematical formulation and its input data need to match, as symmetries in the data (e.g. fully balanced impedance matrices, balanced loads), may imply algorithmic challenges. Symmetries may cause convergence issues [7], through the existence of multiple solutions with the same objective value. Symmetry breaking can be crucial to improving the reliability and performance of algorithms. It is important not to tightly couple the general engineering data model to a small number of mathematical formulations, as the goal is to use the datasets for benchmarking across many formulations and solvers. We note that UBOPF in many ways is just a starting point for the development in this field, not the end goal. UBOPF has been shown to be a good foundation to develop approaches in distribution network expansion planning, distribution state estimation [8] and parameter estimation [9], multiperiod economic dispatch [10] and more. By developing comprehensive, complete, and extensible data models now, we hope to accelerate the development of new methods across a wide variety of distribution network problems. For instance convex relaxation techniques applied to OPF have pushed the scalability of global optimizers [11].

Third, to further the uptake of the data sets in the academic community, it is important to not overcomplicate the data models. Nevertheless, existing data models (e.g. in OpenDSS) often force the user to decompose uncommon components such as split-phase transformers or open-delta regulators into primitive building blocks, potentially leading to compromised computational performance. Thus, the engineering data model design should avoid such default decompositions.

B. Scope and contributions

Given these observations, it is clear that a range of considerations must be addressed when defining a data model for UBOPF and its extensions. To systematically approach this task, and precisely outline the scope of this paper, we derive a layered *architecture* for handling the data sources and transformations that occur within a UBOPF (or any OPF) study workflow. This architecture is illustrated in Fig. 1, and is described in more detail in the next section, which also reviews the necessary background material and related work.

C. Paper structure

Building on the architecture, the paper then progresses as follows: Section II reviews the literature on data architectures for benchmarking. Section III enumerates the design goals of a UBOPF data model, by working through the pitfalls of existing benchmark test cases and drawing out implied assumptions that are sometimes overlooked. These assumptions and pitfalls must be overcome to enable fair benchmarking, and avoiding them is a major justification for our proposed data model specifications. Section IV describes our proposal for tiered engineering data model specifications and their associated data sets. This section focuses on the lowest tier and show how to define the data model in a way that enables rapid benchmarking of research-grade unbalanced OPF implementations fit for a variety of purposes. Using the tier 1 data model, Section V illustrates the pitfalls of (inadvertently) poor data model choices with numerical examples. Section VI discusses the roadmap for higher tier data models and presents the conclusions.

II. DATA ARCHITECTURE AND RELATED WORK

We propose a data architecture that takes inspiration from approaches in software engineering, data science and machine learning, where ongoing operations are organized according to formal practices and processes that improve ease of use and efficiency [12]; these are variously called DevOps, DataOps and MLOps (*machine learning* ops). The closest to our proposed architecture is MLOps [13], which defines a set of practices and tools that streamline and automate the deployment, monitoring, and management of new ML systems and analytics platforms in production environments. These tools aims to bridge the gap between data science and information technology operations, ensuring that new computational systems, typically developed through research endeavours, can be deployed, maintained, and scaled effectively to deliver value to organizations.

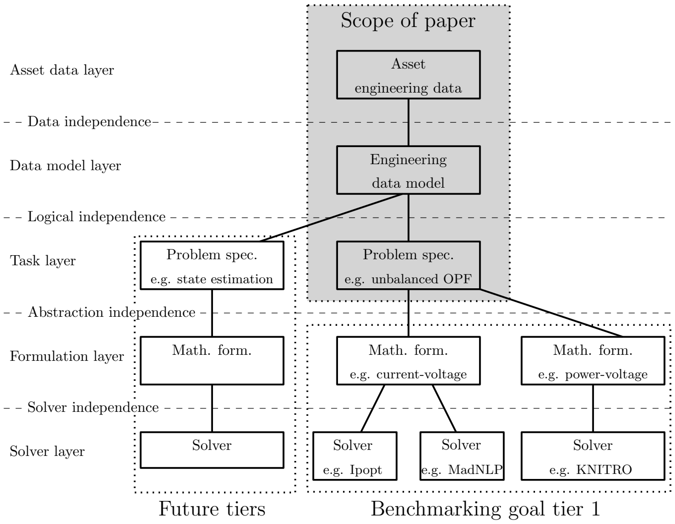


Fig. 1. Diagram illustrating the scope of work.

A. UBOPF benchmarking architecture

In this paper, we advance a similar long term goal — to facilitate the seamless deployment, monitoring, and management of data-driven UBOPF models, ensuring their reproducibility, reliability, scalability, and ongoing performance in real-world applications. The starting point is our proposed data architecture, which comprises five layers in a hierarchy and four interfaces between them, as shown in Fig 1. The layers are:

- **Asset data layer**: The raw asset engineering data collected and stored by asset management teams in utilities.
- **Data model layer**: An engineering data model describing rules or conventions that govern how asset data is to be stored and datasets abiding these rules.
- **Task layer**: A description of a specific optimisation problem or simulation task to be conducted utilising the data in the data model. The problem specifications include physical and operational constraints as well as objectives for optimisation problems.
- **Formulation layer**: A concrete instantiation of a problem specification abiding the mathematical characteristics of certain mathematical programming abstractions, or optimisation problem classes, including approximations and relaxations as well as exact formulations.
- **Solver layer**: Various mathematical programming solvers or other algorithms for solving different classes of optimisation problems.

Between the layers we have the interfaces that define forms of independence separating the layers. Indeed, it could be argued that the points at which independences can be identified actually define layers. These interfaces are:

- **Data independence:** The form of the asset data is independent of the engineering data model. The data model can change, but the asset engineering datasets stay the same. Note that in practice, the asset data may be stored in very different ways across organisations, but ideally any complete dataset can be transformed into the same, consistent, data model. Also note that we propose a hierarchy of data models, rather than one canonical, complete data model.
- **Logical independence**: The data model is independent of the problem specification. The problem specification can change but the data model stays the same. Importantly, this allows for consistent evaluations across different problem specifications.
- **Abstraction independence**: The problem specification is independent of the mathematical formulation. The mathematical formulation can change but the problem specification stays the same.
- **Solver independence**: Solver is independent of mathematical abstraction. Solvers can change but the abstraction stays the same. This layer is well-defined in the state of the art and is typically handled by toolboxes such as JUMP, PYOMO, AMPL and GAMS.

With reference to Fig 1, the scope of this paper is the top three layers: asset data layer, data model layer and task layer, focusing on data models in particular. Our broad aim is to use this architecture to support accelerated research into UBOPF and related tasks, by systematically organising datasets using a tiered hierarchy of data models. To this end, we illustrate the value of adopting such a structure data model on a UBOPF under two formulations (formulation layer) and using three solvers (solver layer), which also serve to highlight some common pitfalls in UBOPF modelling.

As this work does not stand in isolation, we now review related work on benchmarking and test case datasets.

B. Benchmarking power network optimization

The IEEE PES Task Force on algorithmic benchmarking exists to foster the development of more and more interesting data sets for comparing different power system *optimization* methods [14]. This includes optimal power flow (OPF) and unit commitment focused on transmission networks, but the task force invites contributors to develop new data sets and problem specifications, including those for distribution networks. In the optimization context, the focus of *benchmarking* is slightly different from simulation validation. The goal is to not just validate power flow feasibility (which can be handled through existing simulation engines), but also to: (i) establish best-known optimal solutions to nonconvex problems, and (ii) find these solutions quickly and reliably across a wide range of test cases. Identifying scalable approaches will enable the development of the next generation of algorithms, e.g. to solve large-scale security-constrained OPF problems [15]. Benchmarking has been crucial to establish evidence of the power-voltage (polar) formulation being the most performant for transmission networks [16].

C. Availability of distribution network data

In the distribution context, the IEEE PES Test Feeder working group has developed test models[1] for more than 20 years [17]. These test feeders provide a set of distribution system models to *validate* implementations of new distribution system *simulation* methods. Therefore, the test models include meshed topologies, electrically and physically parallel transformers and cables, rare transformer designs, ‘exponential’ loads, harmonics, short circuits, and more. These network models are used to (i) ensure simulation results match across different tools, (ii) identify which components are supported accurately or through approximation, and/or (iii) determine algorithms’ reliability and scalability.

In the IEEE test feeders, exotic components or network configurations may be over-represented relative to real-world networks, as the goal was to push power flow simulation engines to their limits. There is a need for libraries of *representative* networks as well, enabling researchers to test and analyse new technologies in realistic simulation-based studies.

III. CONSIDERATIONS AND DESIGN GOALS

In general, there is a tension between the expressivity of the data model (what it can represent) and the computational complexity of the problems at hand. Although desirable, more detailed data models imply a greater degree of detail of the system under study that entails increased computational burden, as well as complexity for researchers in handling and processing the data. As such, research activities should be verifiable and replicable, so excess complexity should be avoided where possible. On the other hand, failing to include important details in the data model can leave models derived from it with an incomplete representation of reality, thereby limiting industrial uptake. This can result in poor accuracy or alignment with the system under study, as discussed in [18] and illustrated in Section V. The overarching aim of this work is to find a balance between these competing objectives, by reducing the practical complexity of working with UBOPF benchmarks, while retaining expressivity where it is needed.

Throughout this section, we review the state of benchmarking activities for OPF and power flow, and UBOPF specifically. We document lessons learned by previous related benchmark generation activities, and highlight some common flaws in approaches typically used to generate OPF datasets, including storing already-transformed data or data for some assets post-decomposition. Then, based on this review, we lay out our data model design goals and specific data model proposal in subsequent sections.

---

[1] https://cmte.ieee.org/pes-testfeeders/resources/

A. Learning lessons from static transmission benchmarking

A prominent example of quasi-static transmission optimization benchmarking is 'Power Grid Library' (PG Lib) [14], which is supported by open-source software packages: MATPOWER (in MATLAB), PANDAPOWER in Python, and POWERMODELS in Julia.

In the authors' experience, benchmarks released by PG Lib have been relatively successful in achieving uptake by the academic community. PG Lib repositories with benchmark data sets are curated and maintained by the IEEE Power and Energy Society *Task Force on Benchmarks for Validation of Emerging Power System Algorithms*. We believe a number of reasons were critical for its success. PG Lib used a relatively simple data model and data format, based on MATPOWER. The format has a single objective definition — generation cost minimization — easing configuration and use, but tightly coupling the data model and mathematical representation, which are layers we wish to keep independent.

To become part of PG Lib, a number of data standardization steps are performed to ensure the data integrity, including: tagging edges as lines or transformers; automatically deriving additional benchmarks (Active Power Increase / Small Angle Difference); embedding feasible power flow results in the case file, and; overwriting angle difference bounds with reasonable values. In practice, the format is also used as a basis for related problem specifications, such as security-constrained OPF[2] and transmission network expansion planning, or OPF with HVDC systems and multiple asynchronous zones [19]. Such initiatives may eventually be turned into new benchmarking initiatives, e.g. PG-Lib-HVDC[3] and PG-Lib-UC[4] (unit commitment).

However, a few choices in the PG Lib data model are limiting in practice. Only one (constant power) load per bus is allowed, which can be cumbersome for problem specifications where fixed and flexible loads, or storage, need to co-exist. There is no support for three-winding transformers, forcing decomposition into multiple two-winding ones. Switches and zero impedance sections are not supported. Load models are limited to constant power and constant current. Flow limits are either in power or in current, mixing of bounds is not supported. Finally, we point the readers to J.D. Lara et al. who discuss the design of an open-source *management tool* for transmission network data [20].

B. Learning lessons from distribution simulation tools

OPENDSS is a widely-used open-source unbalanced network *simulation* tool used to evaluate network dynamics. OPENDSS' data model is very flexible, for instance with reusable definitions of transformer and line parameters (a.k.a 'construction codes', or 'line codes'). Decomposing the building of impedances into a) per-length impedances specified by construction codes, and b) lengths for specific sections, is common practice in the development of distribution network models. These per-length impedances can be set up in different ways: a) the modeler fills out impedance values (e.g. phase-coordinate or sequence coordinate values) or b) set up the inputs to Carson's equations[5]. For the latter, OPENDSS derives the impedance values before running the power flow.

The flexibility of OPENDSS' data model, while powerful, comes with disadvantages for modelers as well. It is more difficult to come up with rules to check the data for consistency, and alert the user to issues before calling the solver. The data model even supports the mixing of Kron-reduced and explicit-neutral line models in a single case study. This is challenging from the perspective of debugging data sets. Furthermore, it is often not obvious how to set up the data to represent components such as split-phase transformers or open-delta regulators from nameplate properties of such components.

C. Maximizing community uptake

To further the uptake of the data sets in the academic community, it is important to not overcomplicate the data models. Nevertheless, existing data models (e.g. in OPENDSS) often force the user to decompose uncommon components such as split-phase transformers or open-delta regulators into primitive building blocks. Undoing such decompositions is tricky, but may be beneficial for computational performance. Thus, the engineering data model design should avoid such default decompositions.

Note that the broader ecosystem and licensing are crucial in enabling the community to do more benchmarking:

- the data license needs to enable adaptation and attribution, e.g. creative commons in PG-Lib OPF.
- a reference mathematical specification, e.g. as on the landing page at [6].
- data model specification, e.g. MATPOWER manual[7]
- example model building scripts with permissive license, e.g., ROSETTA-OPF[8].
- scientific toolboxes with permissive licenses, e.g. POWERMODELSDISTRIBUTION.
- a serialization choice that enables data parsing across a variety of languages, e.g. JSON.
- community members should be able to contribute new data sets, e.g. through pull requests on GITHUB.

Note that the goal is not to replace or compete with common information model (CIM), which has a clear purpose in enabling the integration of *enterprise* software systems.

D. Describing physical assets, not transformations

Engineering data models capture the essential features of the data describing the system under consideration. Such data model is then compiled into input data to instantiate a concrete mathematical model for use by a particular algorithm. Through our examples we show why it is good practice to keep these data-handling steps distinct. The seminal PANDAPOWER

---

[2] https://github.com/lanl-ansi/PowerModelsSecurityConstrained.jl
[3] https://github.com/power-grid-lib/pglib-opf-hvdc
[4] https://github.com/power-grid-lib/pglib-uc
[5] distances between conductors, material types, cross sections
[6] https://github.com/power-grid-lib/pglib-opf
[7] https://matpower.org/doc/
[8] https://github.com/lanl-ansi/rosetta-opf/blob/main/jump.jl

paper argues for [21] *element-based* models: "…defining the network with nameplate parameters, such as length and relative impedance for lines, or short circuit voltage and rated apparent power for transformers."

Note that this perspective aligns with how commercial asset management divisions approach the development of asset databases. The goal is to describe the properties of the asset, not the properties of specific technical representations of the asset. For instance, it is fairly obvious when an overhead line section is four-wire or three-wire. Furthermore, from first principles, e.g. using Carson's equations and information on geometry and materials, we know that line impedances of four-wire lines are modeled with 4×4 matrices.

In this context, computational tools can choose to further transform the asset properties into different representation, e.g. 3×3 matrix with a Kron-reduced neutral. However, as many of such transformations are not bijective or reversible, model developers need to pay attention to potential loss of information when (only) storing different electrical engineering representations of properties such as impedance.

This approach also aligns with separation of concerns between i) describing assets using real-world data models, and ii) solving simulation and optimization models, whether exact or approximate, using that established data. Asset modelers should *not* be *power system* modeling experts, as collecting physical features and spec sheets should not require such skills.

One of the choices in the data model is that of the line model. Fig. 2 illustrates a four-wire Π section. Some tools only allow the matrix entries to be defined indirectly, e.g. in symmetrical components, or assume diagonal shunt admittance matrices. We choose not to make any of these assumptions.

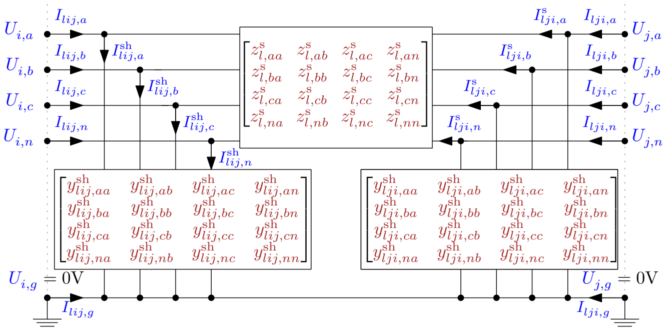


Fig. 2. Four-wire Π line model.

E. Data model must suit real world needs

Power system data models come in different forms:

- node-breaker model (how the data is stored in EMS);
- bus-branch model (interface to most power flow tools/analytics/engines); and
- models with fixed versus variable amounts of wires.

Often, node-breaker models, typically containing various zero impedance components, are converted to the bus-branch models that typical engines accept, thereby also eliminating zero impedance sections. However, converting the primary engineering data to remove these elements on ingestion to the data store, limits the use of the same data sets for other more complicated UBOPF problems requiring richer data. For example, this approach is impossible when the problem statements include switch variables, e.g. optimal network switching, or state estimation methods to figure out unreported switching actions. Instead, by adding (optional) switch components to the data model, both node-breaker and bus-branch models can be represented in the same data model.

IV. Mathematical Specification and Data Model

Conceptually, we establish problem specifications independently of their mathematical representation. In this section, by necessity of making our work practical, we establish a reference mathematical formulation for the four-wire UBOPF generation cost minimization problem, that allows for reduced amounts of terminals for components, e.g. single-phase two-wire lines and connected loads. The current-voltage formulation i) is an exact model for the electrical physics[9] ii) and it can be lifted to the power-voltage (the polar or rectangular forms) and power-lifted-voltage (e.g. used in BIM/BFM SDP relaxations) variable spaces using established techniques [22]. We note that lifting may introduce spurious solutions in the absence of voltage magnitude lower bounds [6], [3].

A. Tier 1 mathematical model

1) `bus`*:* The set of nodes is $\mathcal{N} = \{a, b, c, n\}$, and the subset $\mathcal{P} = \{a, b, c\}$ are called the phases. Voltage-to-ground variables at bus $i \in \mathcal{I}$ are,

$$\mathbf{U}_i \overset{\text{def}}{=} \begin{bmatrix} U_{i,a} & U_{i,b} & U_{i,c} & U_{i,n} \end{bmatrix}^{\text{T}}, \quad U_{i,g} = 0 \text{ V}. \tag{1}$$

The bounds on voltage-to-ground magnitude are,

$$\mathbf{U}_i^{\text{min}} = \begin{bmatrix} U_{i,a}^{\text{min}} \\ U_{i,b}^{\text{min}} \\ U_{i,c}^{\text{min}} \\ 0 \end{bmatrix} \leq \begin{bmatrix} |U_{i,a}| \\ |U_{i,b}| \\ |U_{i,c}| \\ |U_{i,n}| \end{bmatrix} \leq \begin{bmatrix} U_{i,a}^{\text{max}} \\ U_{i,b}^{\text{max}} \\ U_{i,c}^{\text{max}} \\ U_{i,n}^{\text{max}} \end{bmatrix} = \mathbf{U}_i^{\text{max}} \tag{2}$$

We can write this in a compact and differentiable manner,

$$\mathbf{U}_i^{\text{min}} \circ \mathbf{U}_i^{\text{min}} \leq \mathbf{U}_i \circ \mathbf{U}_i^{*} \leq \mathbf{U}_i^{\text{max}} \circ \mathbf{U}_i^{\text{max}}. \tag{3}$$

The phase-to-neutral voltages are a linear combination of the phase-to-ground voltages,

$$\mathbf{U}_i^{\perp} \overset{\text{def}}{=} \begin{bmatrix} U_{i,an} \\ U_{i,bn} \\ U_{i,cn} \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} U_{i,a} - U_{i,n} \\ U_{i,b} - U_{i,n} \\ U_{i,c} - U_{i,n} \end{bmatrix} \tag{4}$$

The phase-to-neutral voltage magnitude bounds are,

$$\mathbf{U}_i^{\perp,\text{min}} \circ \mathbf{U}_i^{\perp,\text{min}} \leq \mathbf{U}_i^{\perp} \circ (\mathbf{U}_i^{\perp})^{*} \leq \mathbf{U}_i^{\perp,\text{max}} \circ \mathbf{U}_i^{\perp,\text{max}}. \tag{5}$$

Kirchhoff's current law at bus i is,

$$\forall i: \underbrace{\sum_{lij} \mathbf{I}_{lij}}_{\text{lines}} + \underbrace{\sum_{wij} \mathbf{I}_{wij}}_{\text{switches}} + \underbrace{\sum_{id} \mathbf{I}_{d}}_{\text{loads}} - \underbrace{\sum_{ig} \mathbf{I}_{g}}_{\text{generators}} + \underbrace{\sum_{ih} \mathbf{Y}_{h} \mathbf{U}_i}_{\text{shunts}} = 0 \tag{6}$$

---

[9] assuming steady-state

The respective current vectors are,

$$\mathbf{I}_{lij} \overset{\text{def}}{=} \begin{bmatrix} I_{lij,a} \\ I_{lij,b} \\ I_{lij,c} \\ I_{lij,n} \end{bmatrix}, \mathbf{I}_{wij} \overset{\text{def}}{=} \begin{bmatrix} I_{lij,a} \\ I_{lij,b} \\ I_{lij,c} \\ I_{lij,n} \end{bmatrix}, \mathbf{I}_{d} \overset{\text{def}}{=} \begin{bmatrix} I_{d,a} \\ I_{d,b} \\ I_{d,c} \\ I_{d,n} \end{bmatrix}, \mathbf{I}_{g} \overset{\text{def}}{=} \begin{bmatrix} I_{g,a} \\ I_{g,b} \\ I_{g,c} \\ I_{g,n} \end{bmatrix}.$$

2) `linecode`*:* For a line l, we derive the line impedance from the linecode c, by multiplying the per-length impedance with the line length ℓ_l.

$$\mathbf{Z}^{\text{s}}_{l} \leftarrow \mathbf{Z}^{\text{s}}_{c} \cdot \ell_l, \quad \mathbf{Y}^{\text{sh}}_{lij} \leftarrow \mathbf{Y}^{\text{sh}}_{c} \cdot \ell_l/2, \quad \mathbf{Y}^{\text{sh}}_{lji} \leftarrow \mathbf{Y}^{\text{sh}}_{c} \cdot \ell_l/2 \quad (7)$$

Linecode current limits are copied to line current limits,

$$\mathbf{I}^{\text{max}}_{l} \leftarrow \mathbf{I}^{\text{max}}_{c} \quad (8)$$

3) `line`*:* The terminal currents flowing into a line l, at bus i in the direction of bus j is $\mathbf{I}_{lij}$. The current magnitude bounds are,

$$\begin{bmatrix} |I_{lij,a}| \\ |I_{lij,b}| \\ |I_{lij,c}| \\ |I_{lij,n}| \end{bmatrix} \leq \begin{bmatrix} I^{\text{max}}_{l,a} \\ I^{\text{max}}_{l,b} \\ I^{\text{max}}_{l,c} \\ I^{\text{max}}_{l,n} \end{bmatrix} = \mathbf{I}^{\text{max}}_{l} \quad (9)$$

A differentiable and convex expression for these bounds is,

$$\mathbf{I}_{lij} \circ (\mathbf{I}_{lij})^{*} \leq \mathbf{I}^{\text{max}}_{l} \circ \mathbf{I}^{\text{max}}_{l}, \mathbf{I}_{lji} \circ (\mathbf{I}_{lji})^{*} \leq \mathbf{I}^{\text{max}}_{l} \circ \mathbf{I}^{\text{max}}_{l}. \quad (10)$$

The current flowing into the line divides over the series ($\mathbf{I}^{\text{s}}_{lij}$) and shunt ($\mathbf{I}^{\text{sh}}_{lij}$) elements,

$$\mathbf{I}_{lij} = \mathbf{I}^{\text{sh}}_{lij} + \mathbf{I}^{\text{s}}_{lij}. \quad (11)$$

We can substitute in the shunt admittance,

$$\mathbf{I}_{lij} = \mathbf{Y}^{\text{sh}}_{lij} \mathbf{U}_{i} + \mathbf{I}^{\text{s}}_{lij}. \quad (12)$$

At the receiving and we get a similar expression,

$$\mathbf{I}_{lji} = \mathbf{Y}^{\text{sh}}_{lji} \mathbf{U}_{j} + \mathbf{I}^{\text{s}}_{lji}. \quad (13)$$

Ohm's law over line l, between buses i and j is,

$$\mathbf{U}_{j} = \mathbf{U}_{i} - \mathbf{Z}^{\text{s}}_{l} \mathbf{I}^{\text{s}}_{lij}. \quad (14)$$

Conservation of current allows to eliminate one of the directional series current variables,

$$\mathbf{I}^{\text{s}}_{lij} + \mathbf{I}^{\text{s}}_{lji} = 0 \iff \mathbf{I}^{\text{s}}_{lji} = -\mathbf{I}^{\text{s}}_{lij}. \quad (15)$$

4) `switch`*:* We use the label switch to represent sections with very low impedance relative to the rest of the network, e.g. switches, breakers, busbars or short lines. With impedance going to 0,

$$\mathbf{U}_{j} = \mathbf{U}_{i} - \epsilon \mathbf{Z}^{\text{s}}_{l} \mathbf{I}^{\text{s}}_{lij}, \quad \epsilon \rightarrow 0, \quad (16)$$

the mathematical model for the closed switch is lossless,

$$\mathbf{I}_{wij} + \mathbf{I}_{wji} = 0, \mathbf{U}_{i} = \mathbf{U}_{j} \implies \mathbf{S}_{wij} + \mathbf{S}_{wji} = 0. \quad (17)$$

For an open switch, $\mathbf{I}_{wij} = \mathbf{I}_{wji} = 0$ and voltages are unlinked. Current limits are,

$$\mathbf{I}_{wij} \circ (\mathbf{I}_{wij})^{*} \leq \mathbf{I}^{\text{max}}_{w} \circ \mathbf{I}^{\text{max}}_{w}. \quad (18)$$

5) `voltage_source`*:* To establish a unique solution, one voltage angle needs to be specified, relative to which all other voltage angles can be interpreted. At specific buses, and at least one for each contiguous network section, we may specify a fixed bus voltage phasor s,

$$\mathbf{U}_{i} \overset{\text{def}}{=} \begin{bmatrix} U_{i,a} \\ U_{i,b} \\ U_{i,c} \\ U_{i,n} \end{bmatrix} = \begin{bmatrix} U^{\text{ref}}_{s,a} \\ U^{\text{ref}}_{s,b} \\ U^{\text{ref}}_{s,c} \\ U^{\text{ref}}_{s,n} \end{bmatrix} \overset{\text{def}}{=} \mathbf{U}^{\text{ref}}_{s}, \text{e.g.,} \begin{bmatrix} 1\angle 0 \\ 1\angle -2\pi/3 \\ 1\angle 2\pi/3 \\ 0 \end{bmatrix} pu \quad (19)$$

In the context of distribution networks, the term 'infeeder' is also used by practitioners to refer to the reference bus with the slack generator. Next to the infeeder, sometimes practitioners perform perfect neutral grounding at multiple buses through a single network. Note that 0 impedance would result in infinity in the admittance $\mathbf{Y}_{h}$ as used in (6). Instead, perfect grounding is modeled as a voltage source constraint,

$$U_{i,n} = U^{\text{ref}}_{s,n} = 0 \text{ V}. \quad (20)$$

6) `load`*:* For loads d, we define the power complex variables in 'wye',

$$\mathbf{S}^{\text{⅄}}_{d} \overset{\text{def}}{=} \begin{bmatrix} U_{i,a} - U_{i,n} \\ U_{i,b} - U_{i,n} \\ U_{i,c} - U_{i,n} \end{bmatrix} \circ \begin{bmatrix} I_{d,a} \\ I_{d,b} \\ I_{d,c} \end{bmatrix}^{*} \overset{\text{def}}{=} \begin{bmatrix} S_{d,aa} - S_{d,na} \\ S_{d,bb} - S_{d,nb} \\ S_{d,cc} - S_{d,nc} \end{bmatrix}. \quad (21)$$

Note that $S_{d,na}, S_{d,nb}, S_{d,nc}$ are in general nonzero if $U_{i,n}$ is nonzero. Load set points $\mathbf{S}^{\text{ref},\text{⅄}}_{d}$ are defined between phase and neutral,

$$\mathbf{S}^{\text{⅄}}_{d} \overset{\text{def}}{=} \mathbf{P}^{\text{⅄}}_{d} + j\mathbf{Q}^{\text{⅄}}_{d} = \mathbf{S}^{\text{ref},\text{⅄}}_{d} \overset{\text{def}}{=} \mathbf{P}^{\text{ref},\text{⅄}}_{d} + j\mathbf{Q}^{\text{ref},\text{⅄}}_{d}. \quad (22)$$

7) `generator`*:* The power for a wye-connected generator $\mathbf{S}^{\text{⅄}}_{g} \overset{\text{def}}{=} \mathbf{P}^{\text{⅄}}_{g} + j\mathbf{Q}^{\text{⅄}}_{g}$ is,

$$\mathbf{S}^{\text{⅄}}_{g} \overset{\text{def}}{=} \begin{bmatrix} U_{i,a} - U_{i,n} \\ U_{i,b} - U_{i,n} \\ U_{i,c} - U_{i,n} \end{bmatrix} \circ \begin{bmatrix} I_{g,a} \\ I_{g,b} \\ I_{g,c} \end{bmatrix}^{*} \overset{\text{def}}{=} \begin{bmatrix} S_{g,aa} - S_{g,na} \\ S_{g,bb} - S_{g,nb} \\ S_{g,cc} - S_{g,nc} \end{bmatrix} \quad (23)$$

Generators are dispatched between their lower $\mathbf{P}^{\text{min}}_{g}$ and upper $\mathbf{P}^{\text{max}}_{g}$ active power bounds,

$$\mathbf{P}^{\text{min}}_{g} \leq \mathbf{P}^{\text{⅄}}_{g} \leq \mathbf{P}^{\text{max}}_{g}, \quad (24)$$

and between their lower $\mathbf{Q}^{\text{min}}_{g}$ and upper $\mathbf{Q}^{\text{max}}_{g}$ reactive power bounds,

$$\mathbf{Q}^{\text{min}}_{g} \leq \mathbf{Q}^{\text{⅄}}_{g} \leq \mathbf{Q}^{\text{max}}_{g}. \quad (25)$$

8) Objective: We define an objective, using linear costs for power generation by phase,

$$\mathbf{C}_{g} = \begin{bmatrix} C_{g,a} & C_{g,b} & C_{g,c} \end{bmatrix}^{\text{T}}.$$

Note that the costs are generally the same by phase. The expression for the objective now becomes,

$$\min \sum_{g} (\mathbf{C}_{g})^{\text{T}} \Re(\mathbf{S}^{\text{⅄}}_{g}). \quad (26)$$

Note that we absolutely do *not* mean to imply that (linear) generation cost minimization is a realistic problem statement in distribution networks. However, we start with this in Tier 1, as it is i) the most direct generalization of the existing PG Lib benchmark, and ii) easy to implement. The next tiers develop more advanced expressions for objectives, even including nonlinear terms, e.g., curtailment minimization, self-consumption maximization or electricity bill minimization.

9) Summary of feasible set: Table I summarizes the feasible set of the optimization model. This includes the list of expected physical units for the data inputs. Note that we do not specify outputs or how the computation is performed.

B. Tier 1 data model implementation

We propose a set of tiered data models with appropriate tradeoffs between simplicity and representativeness for the task at hand, which enable consistency checks while avoiding ambiguity, semantic abuse, and forced approximation. The proposed Tier 1 data model supports extensions, including for existing entities, e.g. add a new property to an existing load to indicate that it is flexible. We choose to support,

- electrically parallel lines (i.e. defined by triples lij);
- meshed networks (i.e. data model does not define a hierarchy, although it can be recovered uniquely for radial networks);
- 1 to 4-wire line models, both Kron-reduced or with explicit neutral;
- perfect neutral grounding, as well as through impedances;
- constant power, wye loads only.

1) Exclusions for simplicity: We choose not to support a number of features (yet), as they cause a nontrivial amount of implementation overhead. We do not:

- separate connectivity from the asset (like in CIM/PSS-Sincal) with ‘connectivitynodes’;
- allow for two wires to connect to a single node;
- support more than four wires per line;
- force users to fill out small values for short sections;
- support defining impedance data in sequence coordinates;
- voltage source is a fully determined phasor, this could be made less restrictive.
- support internal impedance for the voltage source.

Finally, we do not associate geolocation or visualization data, and leave that for future work. We make no assumptions on hierarchy or orientation of elements in the network, even though this is possible when the network is radial.

TABLE I

FEASIBLE SET OF CURRENT-VOLTAGE UBOPF

Variables	Voltage, current	$\mathbf{U}_i, \mathbf{I}^{\mathrm{s}}_{lij}$
Bounds	Phase-to-ground voltage	(2)/(3)
	Phase-to-neutral voltage	(5)
	Line current	(10)
	Generator dispatch	(24), (25)
Constraints	Ohm’s law	(12), (13), (14)
	Bus KCL	(6)
	Generator power	(23)
	Load power	(21)
	Switch (closed)	(17)
	Voltage source	(19)
	Neutral grounding	(20)

2) Connections to Node Mapping: The default connectivity for four-wire lines or wye loads/generators is,

$$\texttt{connection} \rightarrow \begin{bmatrix}\texttt{1} & \texttt{2} & \texttt{3} & \texttt{4}\end{bmatrix}^{\mathrm{T}},$$

i.e. the first terminal connects to node 1 (a), second to 2 (b), third to 3 (c) and fourth to 4 (n), i.e. following OPENDSS’ de facto convention.

It also creates data validation obstacles; for example, when a single-phase load is fed through a 2-wire line, talking about 0 power consumption on phases that are not physically present is unnecessarily confusing. Therefore, instead we prefer to not have 0 entries for missing phases, but maintain an independent mapping. We define the typical labels for the nodes of a bus, i.e. $\mathcal{N} = \{a, b, c, n\}$. Components terminals connect to the nodes of the bus, and we therefore map onto $\mathcal{N}$.

It is possible to represent single-phase loads,

$$\mathbf{S}_d^{\text{ref},⅄} = \begin{bmatrix} 0 \text{ kW} + j0 \text{ kvar} \\ 2 \text{ kW} + j1 \text{ kvar} \\ 0 \text{ kW} + j0 \text{ kvar} \end{bmatrix}. \tag{27}$$

However, such a data model creates ambiguity of single-phase loads w.r.t three-phase loads with unequal set points. Therefore, for a single-phase load we preferably write,

$$\mathbf{N}_d = \begin{bmatrix} b \\ n \end{bmatrix}, \mathbf{S}_d^{\text{ref},⅄} = \begin{bmatrix} 2 \text{ kW} + j1 \text{ kvar} \end{bmatrix}, \tag{28}$$

to indicate the first terminal connects to b and the second to n, and that the (constant-power) setpoint is $2 \text{ kW} + j1$ kvar. For generators we define the map, $\mathbf{N}_g$. For single-phase lines two-wire, we write,

$$\mathbf{N}_{lij} \rightarrow \begin{bmatrix} b \\ n \end{bmatrix}, \mathbf{N}_{lji} \rightarrow \begin{bmatrix} b \\ n \end{bmatrix}. \tag{29}$$

A neutral grounding shunt can be defined,

$$\mathbf{Y}_h = \begin{bmatrix} Y_{h,nn} \end{bmatrix}, \mathbf{N}_h = \begin{bmatrix} n \end{bmatrix}. \tag{30}$$

Fig. 3 illustrates conservation of current at a bus with a single-phase load and generator, and a neutral grounding shunt.

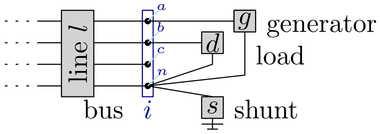


Fig. 3. Illustration of a four-wire line connected to a single-phase generator g, load d and neutral grounding shunt s.

3) JSON file format: JSON can be easily parsed from different programming languages. JSON is based on key-value pairs, and the keys are not ordered. Data extensions into an existing model are performed by adding nested entries. For compatibility across programming languages, we define quantities in the reals, not complex numbers.

4) Nested data model: Conversely to OPENDSS, we define an explicit list of buses, so that each can have their own voltage bounds and other associated properties. Conversely to MATPOWER, we define *unique names* (ids, as strings) for buses, generators, loads, etc. (i.e., we do not force (sequential) integers). The nested data model is organized as follows:

- `name` → `example_case`
- `bus` →
  - $i \in \mathcal{I} \rightarrow$
    - `vmin` → $\mathbf{U}_i^{\text{min}}$ (kV)
    - `vmax` → $\mathbf{U}_i^{\text{max}}$ (kV)
    - `vpnmin` → $\mathbf{U}_i^{\Yup,\text{min}}$ (kV)
    - `vpnmax` → $\mathbf{U}_i^{\Yup,\text{max}}$ (kV)
  - $j \in \mathcal{I} \rightarrow$
    - ...
- `line` →
  - $l \in \mathcal{L} \rightarrow$
    - `length` → ℓ_l (km)
    - `linecode` → c
    - `f_bus` → i
    - `t_bus` → j
    - `f_connections` → $\mathbf{N}_{lij}$
    - `t_connections` → $\mathbf{N}_{lji}$
- `linecode` →
  - $c \in \mathcal{C} \rightarrow$
    - `g_fr` → $\Re(\mathbf{Y}_c^{\text{sh}})$ (S/km)
    - `g_to` → $\Re(\mathbf{Y}_c^{\text{sh}})$ (S/km)
    - `b_fr` → $\Im(\mathbf{Y}_c^{\text{sh}})$ (S/km)
    - `b_to` → $\Im(\mathbf{Y}_c^{\text{sh}})$ (S/km)
    - `rs` → $\Re(\mathbf{Z}_c^{\text{s}})$ (Ω/km)
    - `xs` → $\Im(\mathbf{Z}_c^{\text{s}})$ (Ω/km)
    - `cm_ub` → $\mathbf{I}_c^{\text{max}}$ (A)
    - `is_kron_reduced` → `true` / `false`
- `voltage_source` →
  - $s \in \mathcal{S} \rightarrow$
    - `vm` → $|\circ \mathbf{U}_s^{\text{ref}}|$ (kV)
    - `va` → $\angle \circ \mathbf{U}_s^{\text{ref}}$ (degrees)
    - `bus` → i
    - `connections` →
- `generator` →
  - $g \in \mathcal{G} \rightarrow$
    - `pmin` → $\mathbf{P}_g^{\text{min}}$ (kW)
    - `pmax` → $\mathbf{P}_g^{\text{max}}$ (kW)
    - `qmin` → $\mathbf{Q}_g^{\text{min}}$ (kvar)
    - `qmax` → $\mathbf{Q}_g^{\text{max}}$ (kvar)
    - `cost` → $\mathbf{C}_g$ ($/kWh)
    - `bus` → i
    - `connections` → $\mathbf{N}_g$
- `load` →
  - $d \in \mathcal{D} \rightarrow$
    - `pd_nom` → $\mathbf{P}_d^{\text{ref},\Yup}$ (kW)
    - `qd_nom` → $\mathbf{Q}_d^{\text{ref},\Yup}$ (kvar)
    - `bus` → j
    - `connections` → $\mathbf{N}_d$
- `shunt` →
  - $h \in \mathcal{H} \rightarrow$
    - `g` → $\Re(\mathbf{Y}_h)$ (S)
    - `b` → $\Im(\mathbf{Y}_h)$ (S)
    - `bus` → j
    - `connections` → $\mathbf{N}_h$
- `switch` →
  - $w \in \mathcal{W} \rightarrow$
    - `f_bus` → i
    - `t_bus` → j
    - `cm_ub` → $\mathbf{I}_w^{\text{max}}$ (A)
    - `state` → `open` / `closed`
    - `f_connections` → $\mathbf{N}_{wij}$
    - `t_connections` → $\mathbf{N}_{wji}$

V. ILLUSTRATING PITFALLS IN UBOPF DATA MODELS

We use POWERMODELSDISTRIBUTION [23] to parse the network data and JUMP [24] to build the mathematical model. IPOPT [25] with linear solver MUMPS is used to solve the resulting system of nonlinear equations. Validation of feasibility is performed w.r.t. OPENDSS [26]. Optimality is derived from POWERMODELSDISTRIBUTION [23].

A. Numerical illustrations of pitfalls

1) Small nonzero impedances : When modeling using nodal admittance representations, typically, power engineers replace elements like switches with a very low nominal impedance, that does *not* represent the physical property of the element. The value instead is generally based on how small the impedance needs to be so that the approximation error is 'negligible'. It would be more appropriate to integrate the modeling of low-impedance sections into computational engines, and not force the modeller to come up with values. Furthermore, specialized engines may choose to model such sections using true zero impedance values [27], [28], [6]. Table II illustrates how small impedances influence the behavior of algorithms.

TABLE II
IMPACT OF SMALL NONZERO IMPEDANCE BETWEEN GENERATORS 2 AND 4 FOR 2349-BUS RADIAL NETWORK

	original	small impedance
Solve time (s)	5.7665	8.1668
IPOPT iterations (-)	12	16

2) Padding is inefficient: Padding of impedance matrices can slow down NLP methods unnecessarily. For instance, we pad a 2 × 2 impedance into a 3 × 3,

$$\begin{bmatrix} Z^{\text{s}}_{l,aa} & Z^{\text{s}}_{l,ab} \\ Z^{\text{s}}_{l,ba} & Z^{\text{s}}_{l,bb} \end{bmatrix} \rightarrow \begin{bmatrix} Z^{\text{s}}_{l,aa} & Z^{\text{s}}_{l,ab} & 0 \\ Z^{\text{s}}_{l,ba} & Z^{\text{s}}_{l,bb} & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

This still leaves currents unrestrained, and voltages linked, so we need to drop Ohm's law for the missing phases and add,

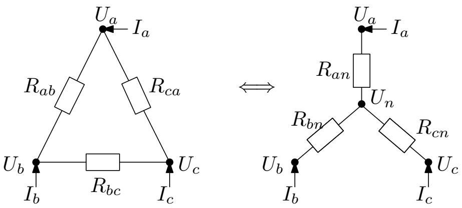
$$I_{lij,n} = 0, I_{lji,n} = 0. \qquad (31)$$

This approach leads to variables for missing wires/nodes that are not necessary, thereby slowing down algorithms, with a numerical example in Table III. Note that computational engines can still choose to do padding, we merely posit that padding in the data (model) should be avoided.

TABLE III
IMPEDANCE MATRIX PADDING COMPUTATIONAL COMPARISON FOR 407-BUS NETWORK

	2 × 2	padded 3 × 3
Objective (-)	23.3533	23.3533
Solve time (s)	1.4278	127.7390
IPOPT iterations (-)	22	590

3) Delta loads cannot simply be converted to wye loads: Delta loads generally can *not* be converted to equivalent wye loads. Wye loads in the context of networks refers to loads connected between phase and neutral. The textbook examples of delta to wye conversion only are exact (from the perspective of the bus supplying the load) when the novel node in the centre of the ‘Y’ is left floating, as can be seen in Fig. 4. Such a load configuration would be rare in real-world public distribution networks; alternatively, connecting the node to the neutral changes the solution.



$$R_{an} = \frac{R_{ab}+R_{ca}}{R_{ab}+R_{bc}+R_{ca}}$$
$$R_{bn} = \frac{R_{bc}+R_{ab}}{R_{ab}+R_{bc}+R_{ca}}$$
$$R_{cn} = \frac{R_{ca}+R_{bc}}{R_{ab}+R_{bc}+R_{ca}}$$
$$I_a + I_b + I_c = 0$$

Fig. 4. Delta to wye conversion as depicted in text books

Table IV lists the voltage and current vectors, in the order $a, b, c, n, star$, for four variants of a two-bus four-wire network with a balanced 3-phase constant-impedance load: i) delta connection, ii) wye with star point connected to *neutral*, iii) wye with star point connected to *ground*, iv) wye with *floating* star point. It is noted only i) and iv) are equivalent per the previously-discussed transformation, the results distinct for the other ones. These small physical differences may cause significant differences in optimality in the presence of voltage or current bounds.

4) Kron's reduction needs to be tracked: When looking at data sets, the following configurations cannot be distinguished based on the size of the impedance matrix:

- 4-wire Kron-reduced;
- 3-wire physical which can represent three-phase, split-phase or two-phase systems.

This limits the ability to do data consistency checks, and therefore also data debugging:

- in 3-phase 3-wire networks, only delta-connected loads / generators / transformers are expected;
- in 4-wire networks, both delta and wye load / generator connections are valid.

TABLE IV

DELTA TO WYE CONVERSION FOR A BALANCED 27 KW + 13.08 J KVAR LOAD PER PHASE ON A FOUR-WIRE LINE IN A 2-BUS SYSTEM.

	Δ	$\curlywedge$-neutral	$\curlywedge$-ground	$\curlywedge$-float
$\lvert\mathbf{I}_d\rvert$ (A)	$\begin{bmatrix}40.607\\41.079\\40.914\end{bmatrix}$	$\begin{bmatrix}40.041\\41.364\\41.185\\1.716\end{bmatrix}$	$\begin{bmatrix}40.528\\41.043\\41.030\end{bmatrix}$	$\begin{bmatrix}40.607\\41.079\\40.914\end{bmatrix}$
$\angle\mathbf{I}_d$ (°)	$\begin{bmatrix}-28.324\\-148.212\\91.160\end{bmatrix}$	$\begin{bmatrix}-28.317\\-148.896\\91.878\\-27.812\end{bmatrix}$	$\begin{bmatrix}-28.200\\-148.368\\91.197\\-33.546\end{bmatrix}$	$\begin{bmatrix}-28.324\\-148.212\\91.160\end{bmatrix}$
$\lvert\mathbf{U}_i\rvert$ (V)	$\begin{bmatrix}215.921\\219.030\\218.915\\3.342\end{bmatrix}$	$\begin{bmatrix}216.145\\218.734\\218.926\\2.676\end{bmatrix}$	$\begin{bmatrix}216.148\\218.893\\218.825\\3.513\end{bmatrix}$	$\begin{bmatrix}215.921\\219.030\\218.915\\3.342\\0.786\end{bmatrix}$
$\angle\mathbf{U}_i$ (°)	$\begin{bmatrix}-2.365\\-122.575\\117.093\\17.379\end{bmatrix}$	$\begin{bmatrix}-2.302\\-122.556\\117.047\\11.662\end{bmatrix}$	$\begin{bmatrix}-2.358\\-122.526\\117.039\\16.866\end{bmatrix}$	$\begin{bmatrix}-2.365\\-122.575\\117.093\\17.379\\143.268\end{bmatrix}$

Note that in most contexts (except for single-wire earth return), we do not expect the ground to serve as the deliberate return path for loads/generators. Therefore, loads/generators connected between a phase and ground in data are unlikely to be physically configured as such. Furthermore, Kron's reduced networks still have valid current bounds on neutral, which we shall now illustrate. We partition $\mathbf{Z}^{\text{s}}_{l}$ and $\mathbf{Y}^{\text{sh}}_{lij}$ on the neutral self-impedance/admittance entry,

$$\mathbf{Z}^{\text{s}}_{l} \overset{\text{def}}{=} \left[\begin{array}{c|c} \mathbf{Z}^{\text{s,PP}}_{l} & \mathbf{Z}^{\text{s,PN}}_{l} \\ \hline \mathbf{Z}^{\text{s,NP}}_{l} & Z^{\text{s}}_{l,nn} \end{array}\right], \mathbf{Y}^{\text{sh}}_{lij} \overset{\text{def}}{=} \left[\begin{array}{c|c} \mathbf{Y}^{\text{sh,PP}}_{l} & \mathbf{Y}^{\text{sh,PN}}_{l} \\ \hline \mathbf{Y}^{\text{sh,NP}}_{l} & Y^{\text{sh}}_{l,nn} \end{array}\right].$$

The expression for the neutral current is,

$$I_{lij,n} = -\frac{1}{Z^{\text{s}}_{l,nn}} \mathbf{Z}^{\text{s,NP}}_{l} \mathbf{I}^{\text{s}}_{lij}[\mathcal{P}] + \mathbf{Y}^{\text{sh,NP}}_{l} \mathbf{U}_{i}[\mathcal{P}]. \tag{32}$$

which remains subject to current bound (10). In networks with neutral wires undersized relative to the phase wires, these limits can be binding first. Note that the neutral may carry more current than any of the phase wires during partial reverse flows, e.g. one phase supplying power downstream, the two others in reverse flow. This may lead to the neutral current bound being more restrictive, thereby influencing the optimality, e.g.,

$$\mathbf{I}^{\text{s}}_{lij}[\mathcal{P}] = \begin{bmatrix} 110.37\angle 152.26° \\ 105.15\angle -118.83° \\ 108.30\angle 121.73° \end{bmatrix} A \Rightarrow I_{lij,n} = 114.81\angle -8.82° A.$$

5) Published datasets may be deficient: It is important to establish properties of the data quality of datasets independent of solvers. This requires data models that can be used to disambiguate. For example, IEEE 123 bus 610 has a delta winding with no reference to earth, and therefore has no unique solution, i.e. mathematically degenerate. OPENDSS by default injects an equivalent shunt admittance of 10 kvar capacitive at 345 kV, thereby resolving the rank deficiency of the nodal admittance matrix. Note that in the physical world, voltages will be unique nevertheless due to coupling to earth, e.g. capacitance. Multiple authors have encountered issues with algorithm development due to this feature [6], [29]. Such ad hoc and potentially blackbox[10] data fixes make it hard to validate the results in other tools, and may lead to situations where different solvers may obtain different solutions.

B. Benchmark data and results

Table V illustrates benchmark results for a number of test cases in the open data release. The four-wire current-voltage formulation of Claeys et al. [6] is used to generate the results. We note significant differences in network size and calculation time in the source data versus the reduced networks, in line with [6]. Test cases in JSON, a parser and scripts with the mathematical model are released on GITHUB[11]

---

[10] OpenDSS is open-source, but commercial solvers are typically not.

[11] https://github.com/frederikgeth/UnbalancedOPFData

TABLE V
BENCHMARK RESULTS

Case name	Objective ($/h)	# Bus		Time (s)	
		original	reduced	original	reduced
nw15f2	92.477	2753	191	22.086	0.837
nw6f1	97.195	3316	200	19.594	0.915
nw9f5	104.414	3590	244	19.550	1.071
nw12f1	135.718	3156	245	21.632	0.972
nw15f4	111.718	4769	247	42.170	1.341
nw15f3	143.174	4080	302	56.429	0.977
nw5f3	139.922	2903	315	31.582	1.774
nw17f1	162.847	2843	376	33.556	2.116
nw17f6	190.365	3856	435	29.615	1.982
nw8f2	263.941	7023	538	58.117	3.197

VI. CONCLUSIONS AND FUTURE WORK

We discussed considerations and design goals to enable the community-driven development of benchmarks for unbalanced optimal power flow. We established a data architecture to allow for data re-use and benchmarking across different programming languages, mathematical models and solvers. We illustrated the necessity of maintaining semantics in the data model to make data debugging more convenient, and to avoid benchmarking pitfalls. The long term goal is to enable data releases and benchmarking for a variety of problems that use UBOPF as a building block. More expressive data models require more implementation effort, and we proposed a tiered approach, with a roadmap illustrated in Table VI. Note that other problem statements, e.g. state estimation, are in scope of the roadmap, and fairly trivial extensions of the proposed data model.

TABLE VI
ILLUSTRATIVE ROADMAP FOR FUTURE DATA MODEL TIERS

Tier 1	See §IV, scope of this paper, later tiers are future work
Tier 2	add: delta loads; common transformer types, e.g. Dy11; networks spanning multiple voltage levels; bounds on voltage angle differences between buses; bounds on voltage angle differences on a bus; bounds on the positive, negative and /or zero sequence voltage or current magnitudes; alternative voltage source models; apparent power bounds.
Tier 3	add: ZIP and exponential load models; defining linecodes through Carson's equations; less common transformer types, e.g. SWER isolation; switch states as variables; extending data model with geospatial features.
Tier ∞	multiperiod a.k.a time series data; split-phase, three-phase, single-phase, n-winding Δ⅄Z transformers; storage models; solar systems with Volt-var/Volt-Watt control; voltage regulators including open delta; single-wire earth return isolation transformers; state estimation weighted least squares and other objectives.

In future work we perform broader benchmarking of different formulations, hardware platforms, solvers and/or automatic differentiation engines. We hope our approach will enable the development of better methods for data debugging. We furthermore want this to be a community-driven initiative, and therefore invite interested parties to contact the authors through email or through the issue tracker[12].

ACKNOWLEDGEMENTS

The development team of POWERMODELSDISTRIBUTION established a lot of the lessons learnt which have been detailed in this work, and therefore we thank Sander Claeys, David Fobes and Carleton Coffrin for their contributions. The authors also like to thank all contributors to other open-source packages that were crucial to this work, including but not limited to, JUMP, IPOPT, MATPOWER, and OPENDSS. GridQube-affiliated authors received funding from AusIndustry grant ACGRANT000837.

REFERENCES

[1] S. Claeys, M. Vanin, F. Geth, and G. Deconinck, "Applications of optimization models for electricity distribution networks," *WIREs Energy and Environment*, vol. 10, no. 5, p. e401, 2021.

[2] A. Dubey and S. Paudyal, "Distribution system optimization to manage distributed energy resources for grid services," *Foundations Trends Elect. Energy Syst.*, vol. 6, no. 3-4, pp. 120–264, 2023.

[3] F. Geth, "Pitfalls of zero voltage values in optimal power flow problems," in *IEEE Power Energy Soc. General Meeting, Orlando FL*, 2023, p. 5.

[4] A. J. Urquhart and M. Thomson, "Assumptions and approximations typically applied in modelling LV networks with high penetrations of low carbon technologies," 10 2013.

[5] L. R. De Araujo, D. R. R. Penido, and F. D. A. Vieira, "A multiphase optimal power flow algorithm for unbalanced distribution systems," *Int. J. Elec. Power Energy Syst.*, vol. 53, no. 1, pp. 632–642, 2013.

[6] S. Claeys, F. Geth, and G. Deconinck, "Optimal power flow in four-wire distribution networks: Formulation and benchmarking," *Elect. Power Syst. Res.*, vol. 213, p. 108522, 2022.

[7] L. Liberti, "Symmetry in mathematical programming," vol. 154, 11 2012.

[8] M. Vanin, T. Van Acker, R. D'hulst, and D. Van Hertem, "A framework for constrained static state estimation in unbalanced distribution networks," *IEEE Trans. Power Syst.*, vol. 37, no. 3, pp. 2075–2085, 2022.

[9] M. Vanin, F. Geth, R. D'hulst, and D. Van Hertem, "Combined unbalanced distribution system state and line impedance matrix estimation," *Int. J. Elect. Power Energy Syst.*, vol. 151, p. 109155, 2023.

[10] F. Geth, C. Coffrin, and D. Fobes, "A flexible storage model for power network optimization," in *Proc. ACM Int. Conf. Future Energy Syst.*, ser. e-Energy '20, New York, NY, USA, 2020, p. 503–508.

[11] S. Gopinath, H. Hijazi, T. Weisser, H. Nagarajan, M. Yetkin, K. Sundar, and R. Bent, "Proving global optimality of ACOPF solutions," *Elect. Power Syst. Res.*, vol. 189, p. 106688, 2020.

[12] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Proc. 28th Int. Conf. Neural Information Process. Syst. Vol. 2*. Cambridge, MA, USA: MIT Press, 2015, p. 2503–2511.

[13] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine learning operations (mlops): Overview, definition, and architecture," *IEEE Access*, vol. 11, pp. 31 866–31 879, 2023.

[14] Babaeinejadsarookolaee et al., "The power grid library for benchmarking ac optimal power flow algorithms," *[math.OC]*, pp. 1–17, 2019.

[15] Aravena et al., "Recent developments in security-constrained ac optimal power flow: Overview of challenge 1 in the ARPA-E grid optimization competition," 2022.

[16] J. Kardos, D. Kourounis, O. Schenk, and R. Zimmerman, "Complete results for a numerical evaluation of interior point solvers for large-scale optimal power flow problems," 2020.

[17] Schneider et al., "Analytic considerations and design basis for the ieee distribution test feeders," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 3181–3188, 2018.

[18] W. Kersting, "The whys of distribution system analysis," *IEEE Industry Appl. Mag.*, vol. 17, no. 5, pp. 59–65, 2011.

---

[12] https://github.com/frederikgeth/UnbalancedOPFData

[19] H. Ergun, J. Dave, D. Van Hertem, and F. Geth, “Optimal power flow for ac–dc grids: Formulation, convex relaxation, linear approximation, and implementation,” *IEEE Trans. Power Syst.*, vol. 34, no. 4, pp. 2980–2990, 2019.

[20] J. D. Lara, C. Barrows, D. Thom, D. Krishnamurthy, and D. Callaway, “Powersystems. jl—a power system data management package for large scale modeling,” *SoftwareX*, vol. 15, p. 100747, 2021.

[21] 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 Trans. Power Syst.*, vol. 33, no. 6, pp. 6510–6521, 2018.

[22] L. Gan and S. H. Low, “Convex relaxations and linear approximation for optimal power flow in multiphase radial networks,” in *Power Syst. Comput. Conf.*, 2014, pp. 1–9.

[23] D. M. Fobes, S. Claeys, F. Geth, and C. Coffrin, “PowerModelsDistribution.jl: An open-source framework for exploring distribution power flow formulations,” *Elect. Power Syst. Res.*, vol. 189, p. 106664, 2020.

[24] M. Lubin, O. Dowson, J. D. Garcia, J. Huchette, B. Legat, and J. P. Vielma, “JuMP 1.0: recent improvements to a modeling language for mathematical optimization,” vol. 15, no. 3, pp. 581–589.

[25] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Math. Prog.*, vol. 106, pp. 25–57, 2006.

[26] R. C. Dugan and T. E. McDermott, “An open source platform for collaborating on smart grid research,” in *Proc. IEEE Power Energy Soc. General Meeting*, 2011, pp. 1–7.

[27] C.-W. Ho, A. Ruehli, and P. Brennan, “The modified nodal approach to network analysis,” *IEEE Trans. Circuits Syst.*, vol. 22, no. 6, pp. 504–509, 1975.

[28] G. Cirilo Leandro and T. Noda, “Development of a three-phase power-flow calculation method for distribution systems with automatic handling of arbitrary winding connections of transformers,” *Elect. Eng. Japan*, vol. 216, no. 4, p. e23445, 2023. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/eej.23445

[29] M. Bazrafshan and N. Gatsis, “Comprehensive modeling of three-phase distribution systems via the bus admittance matrix,” *IEEE Trans. Power Syst.*, vol. 33, no. 2, pp. 2015–2029, 2018.