Exploring OPF Feasible Spaces using Nonlinear Programming and Continuation Methods

Mohammad Rasoul Narimani,* Jose A. Madrigal,* Katherine R. Davis,** and Daniel K. Molzahn†

***Abstract*—Optimal power flow (OPF) is a key problem in the operation of electric power systems. This problem optimizes an objective function while satisfying constraints imposed by nonlinear power flow equations and engineering limits. Due to their non-convexity, understanding OPF feasible spaces is essential for improving solution algorithms. This paper presents an approach that employs local solvers and continuation power flow methods to visualize OPF feasible spaces. Starting from a particular feasible point, the approach iteratively solves a modified OPF problem to find an additional feasible point that maximizes the distance from the points computed in previous iterations. Subsequently, continuation power flow is executed between each pair of the computed points to discover additional feasible points. We demonstrate the proposed algorithm on test cases with up to several tens of buses.**

I. Introduction

THE Optimal Power Flow (OPF) problem seeks the operating point which optimizes a specified objective function, often the minimization of generation costs, subject to the power flow equations and engineering limits. Utilizing the nonlinear AC power flow model to accurately represent the steady-state behavior of power grids results in the AC OPF problem, which is non-convex, may have multiple local optima [1], and is generally NP-Hard [2]. The OPF problem is key for efficient and reliable operations, with its solutions carrying significant economic, environmental, and social impacts. Since first formulated by Carpentier in 1962 [3], a wide variety of algorithms have been applied to the OPF problem [4], [5].

An OPF problem's feasible space is the set of all operating points that satisfy the system constraints, including limits on power flows, voltages, and power injections. Since the effectiveness of OPF solution algorithms strongly depends on a problem's feasible space characteristics, analyzing and visualizing OPF feasible spaces is key for algorithmic improvements. Accordingly, there has been significant research on characterizing OPF feasible spaces, e.g., [1], [2], [6]–[11].

Prior research employs several types of methods to compute points in the feasible spaces of OPF problems with applications including the creation of visualizations and the training of data-driven models. We next briefly review related research.

The feasible spaces of certain OPF problems can be calculated analytically. For instance, OPF problems involving two-bus systems have analytical solutions [1], [12]. Leveraging the symmetries within a problem can also lead to the derivation of explicit expressions for the feasible spaces of other OPF problems [13]. However, analytic solutions are only available for a small number of special cases. In more complex and less symmetric systems, numerical methods are required to characterize OPF feasible spaces.

Other related research includes methods for determining the feasibility boundary of the power flow equations, which refers to the set of parameters that render the power flow equations unsolvable under small parameter changes. For instance, [14]–[16] outlines several research efforts to calculate the distance to the power flow feasibility boundary, primarily for voltage collapse studies. These approaches typically identify small regions, often a single point, that exist on the boundary of the feasible space for the power flow equations. A more comprehensive continuation-based technique is introduced in [7]. Starting from a point in the interior of the feasible space, the approach in [7] uses a continuation method to find a point on the power flow feasibility boundary. By freeing a single parameter (such as active power injection at a particular bus), the approach in [7] utilizes continuation to trace curves that lie on the power flow feasibility boundary. Although computationally tractable for large systems, it can be challenging to confirm that the technique in [7] captures the entire feasible space due to certain non-convexities such as disconnected components. Additionally, the method in [7] does not account for all of the OPF problem's inequality constraints.

References [17] and [18] present related algorithms for provably computing the entire feasible spaces of small OPF problems up to a specified discretization tolerance. Specifically, the feasible spaces are computed by discretizing certain inequality constraints of the OPF problem, resulting in a set of power flow equations. At each discretization point, all solutions to the power flow equations are obtained using the Numerical Polynomial Homotopy Continuation (NPHC) algorithm [19]. To enhance computational tractability, "bound tightening" and "grid pruning" algorithms are used in combination with convex relaxation techniques to avoid considering discretization points for which the power flow equations are provably infeasible. Reference [20] uses the method from [17] to characterize non-convexities in the feasible spaces for multiple small OPF problems, identifying combinations of limits on reactive power and voltage magnitudes that tend to be associated with non-convexities.

Recent efforts in dataset generation for OPF and security assessment highlight the growing importance of systematic sampling techniques. For instance, advanced importance sampling has been used to efficiently construct probabilistic security assessment databases by biasing samples toward high-

∗: Department of Electrical and Computer Engineering, California State University Northridge (CSUN). Rasoul.narimani@csun.edu, jose.madrigal.378@my.csun.edu, Support from NSF contracts #2308498 and #2523881.

†: School of Electrical and Computer Engineering, Georgia Institute of Technology. molzahn@gatech.edu. Support from NSF contract #2145564.

∗∗: Electrical and Computer Engineering Department, Texas A&M University. kdavis@tamu.edu.

information regions [21]. Directed-walk methods have been proposed to iteratively push operating points toward small-signal stability boundaries, enabling targeted exploration for both security assessment and controller tuning [22]. More recently, OPF-Learn introduced an open-source framework that applies hit-and-run Monte Carlo sampling over SOCP relaxations to generate representative AC OPF datasets [23]. In parallel, trajectory-unified [24] and quotient gradient system methods [25] have been developed to directly characterize feasible spaces of small OPF problems. Collectively, these studies illustrate sampling and relaxation-based strategies that complement optimization-based exploration techniques for constructing high-quality datasets of OPF feasible spaces.

By appropriately selecting objective functions and constraints, repeated use of nonlinear programming solvers can yield multiple points in OPF feasible spaces. The key idea is to select objective functions that promote points which are far from previously found feasible points. In the most closely related prior work to our approach, reference [21] systematically examines various nonlinear objective functions for exploring OPF feasible spaces. The performance of these functions is compared using a novel exhaustive rejection sampling routine and evaluated with the Hausdorff distance metric. Results from five PGLib test cases [26] are analyzed to assess the functions' effectiveness in navigating the non-convex power flow space.

Leveraging nonlinear programming solvers, as in [21], along with continuation power flow methods, we present a new algorithm for computing OPF feasible spaces. Our algorithm starts by solving the OPF problem to get a single feasible point, and then iteratively solves a modified OPF problem to find new feasible points that are maximally distant from previously computed solutions. While [21] recommend the use of log barrier–style objectives to encourage well-separated feasible points, we instead adopt a simpler distance-maximization approach without log terms. This choice was made because continuation power flow in our framework already plays the role of efficiently filling in the regions between points, thereby reducing the need for additional log-barrier–based spreading. After obtaining a well-spaced set of feasible points, continuation power flow is applied between pairs of points to rapidly identify additional feasible points. In this way, our algorithm combines the even distribution benefits of nonlinear programming solvers with the rapid interpolation capability of continuation power flow, achieving broad feasible space coverage.

This paper is organized as follows. Section II overviews the OPF formulation. Section III presents the proposed algorithm for computing points within an OPF problem's feasible space. Section IV presents numerical results obtained from applying our algorithm to various test cases. Finally, Section V concludes the paper.

II. Overview of the OPF Problem

This section overviews the AC OPF problem. Consider an n-bus system, where $\mathcal{N} = \{1, \ldots, n\}$, $\mathcal{G}$, and $\mathcal{L}$ are the sets of buses, generators, and lines. Let $P_i^d + \mathbf{j}Q_i^d$ and $P_i^g + \mathbf{j}Q_i^g$ represent the active and reactive load demand and generation, respectively, at bus $i \in \mathcal{N}$, where $\mathbf{j} = \sqrt{-1}$. Let $g_{sh,i} + \mathbf{j}b_{sh,i}$ denote the shunt admittance at bus i. Let V_i and θ_i represent the voltage magnitude and angle at bus $i \in \mathcal{N}$. For each generator $i \in \mathcal{G}$, define a quadratic generation cost function with coefficients $c_{2,i} \geqslant 0$, $c_{1,i}$, and $c_{0,i}$. Denote $\theta_{lm} = \theta_l - \theta_m$. Specified upper and lower limits are denoted by $\overline{(\cdot)}$ and $\underline{(\cdot)}$, respectively. Buses $i \in \mathcal{N} \backslash \mathcal{G}$ have generation limits set to zero.

Each line $(l, m) \in \mathcal{L}$ is modeled as a Π circuit with mutual admittance $g_{lm} + \mathbf{j}b_{lm}$ and shunt admittance $\mathbf{j}b_{sh,lm}$. (Our approach is applicable to more general line models, such the MATPOWER [27] model that allows for off-nominal tap ratios and non-zero phase shifts.) Let p_{lm}, q_{lm}, and $\overline{s}_{lm}$ represent the active and reactive power flows and the maximum apparent power flow limit on the line that connects buses l and m.

Using these definitions, the OPF problem is

$$\min \quad \sum_{i \in \mathcal{G}} c_{2i} \left(P_i^g\right)^2 + c_{1i} P_i^g + c_{0i} \tag{1a}$$

$$\text{subject to} \quad (\forall i \in \mathcal{N},\ \forall (l, m) \in \mathcal{L})$$

$$P_i^g - P_i^d = g_{sh,i} V_i^2 + \sum_{\substack{(l,m) \in \mathcal{L} \\ \text{s.t. } l = i}} p_{lm} + \sum_{\substack{(l,m) \in \mathcal{L} \\ \text{s.t. } m = i}} p_{ml}, \tag{1b}$$

$$Q_i^g - Q_i^d = -b_{sh,i} V_i^2 + \sum_{\substack{(l,m) \in \mathcal{L} \\ \text{s.t. } l = i}} q_{lm} + \sum_{\substack{(l,m) \in \mathcal{L} \\ \text{s.t. } m = i}} q_{ml}, \tag{1c}$$

$$\theta_{ref} = 0, \tag{1d}$$

$$\underline{P}_i^g \leqslant P_i^g \leqslant \overline{P}_i^g, \tag{1e}$$

$$\underline{Q}_i^g \leqslant Q_i^g \leqslant \overline{Q}_i^g, \tag{1f}$$

$$\underline{V}_i \leqslant V_i \leqslant \overline{V}_i, \tag{1g}$$

$$\underline{\theta}_{lm} \leqslant \theta_{lm} \leqslant \overline{\theta}_{lm}, \tag{1h}$$

$$p_{lm} = g_{lm} V_l^2 - g_{lm} V_l V_m \cos\left(\theta_{lm}\right) - b_{lm} V_l V_m \sin\left(\theta_{lm}\right), \tag{1i}$$

$$q_{lm} = -\left(b_{lm} + b_{sh,lm}/2\right) V_l^2 + b_{lm} V_l V_m \cos\left(\theta_{lm}\right) - g_{lm} V_l V_m \sin\left(\theta_{lm}\right), \tag{1j}$$

$$\left(p_{lm}\right)^2 + \left(q_{lm}\right)^2 \leqslant \left(\overline{S}_{lm}\right)^2, \tag{1k}$$

$$\left(p_{ml}\right)^2 + \left(q_{ml}\right)^2 \leqslant \left(\overline{S}_{lm}\right)^2. \tag{1l}$$

The objective function (1a) minimizes the active power generation cost. Constraints (1b) and (1c) enforce power balance at each bus. Constraint (1d) sets the angle reference. Constraints (1e)–(1h) limit the active and reactive power generation, voltage magnitudes, and angle differences between connected buses. Constraints (1i)–(1j) relate the voltage phasors and power flows on each line, and (1k)–(1l) limit the apparent power flows into both terminals of each line.

III. Proposed Feasible Space Sampling Algorithm

This section presents our proposed algorithm for sampling points in an OPF problem's feasible space, i.e., computing well-spaced points that satisfy (1b)–(1l). We first describe an optimization formulation for computing multiple feasible points. We next modify this formulation to obtain feasible points that are better spread out. We then overview continuation methods that identify new feasible points on the line between pairs of existing feasible points. Finally, we describe how to combine these techniques into our overall algorithm.

A. Finding Points in the OPF Feasible Space

We modify the objective function of the OPF problem in order to find multiple points in the OPF problem's feasible space. We first solve the original OPF problem in (1) to find a single feasible point. To find additional feasible points, we then repeatedly solve the modified OPF problem in (2) that maximizes the distance between the solution and the feasible points that have been calculated thus far:

$$\begin{aligned} \max \quad & \sum_{i\in\mathcal{G}} \sum_{j\in\mathcal{N}_\mathcal{S}} \left(P_i^g - \hat{P}_j^g\right)^2 + \left(Q_i^g - \hat{Q}_j^g\right)^2 \\ \text{subject to} \quad & \text{(1b)–(1l)}, \end{aligned} \tag{2}$$

where $\mathcal{N}_\mathcal{S}$ denotes the set of previously found feasible points and $\hat{P}_j^g + \mathbf{j}\hat{Q}_j^g$ is the complex power generation associated with the j^{th} feasible point in $\mathcal{N}_\mathcal{S}$. Note that simply maximizing the sum of squared distances between the current solution and the previously found OPF solutions may be insufficient to produce a large set of well-spaced points. While this formulation can yield a point that is farthest in total distance from all prior solutions, such a point may still lie close to a cluster of previously identified solutions. In other words, the algorithm can amalgamate points within specific regions of the space while leaving other regions unexplored, as shown in Fig. 1.

B. Enforce Algorithm to Find Distant Points

Applying (2) alone to find multiple solutions within an OPF feasible space does not necessarily yield points that are well separated from each other. The resulting feasible points may cluster in certain regions, leaving other areas of the feasible space unexplored. This is undesirable when the objective is to obtain a diverse set of solutions that collectively represent the feasible space more uniformly. To address this limitation, we introduce an additional constraint to explicitly promote separation among feasible points. Specifically, let $(\hat{P}_j^g, \hat{Q}_j^g)$ denote a feasible solution that has already been found, and let (P_i^g, Q_i^g) represent the decision variables of the current optimization run. We require that the distance between the variables representing the current candidate point and every previously found solution be no less than a threshold Δ.

Let $\mathcal{S}$ denote the set of feasible points that have already been found and stored. For a new candidate feasible point with decision variables (P_i^g, Q_i^g), we enforce that it must be at least a distance Δ away from every prior point $(P_j^g, Q_j^g) \in \mathcal{S}$:

$$\left(P_i^g - \hat{P}_j^g\right)^2 + \left(Q_i^g - \hat{Q}_j^g\right)^2 \geqslant \Delta, \quad \forall (\hat{P}_j^g, \hat{Q}_j^g) \in \mathcal{S}. \tag{3}$$

Our proposed algorithm iteratively solves (2) augmented with constraint (3) to obtain a new feasible point, adds it to the set $\mathcal{S}$, and updates constraint (3) so that subsequent solutions remain at least Δ away from all prior points.

This procedure distributes feasible points across the solution space, avoiding clustering. The parameter Δ controls spacing: smaller values yield denser but more clustered solutions, while larger values enforce stronger separation at the cost of fewer points. In practice, Δ may be scaled to generator output ranges or tuned empirically. Figs. 1-2 illustrate that enforcing (3) results in a more uniform spread of obtained feasible points.

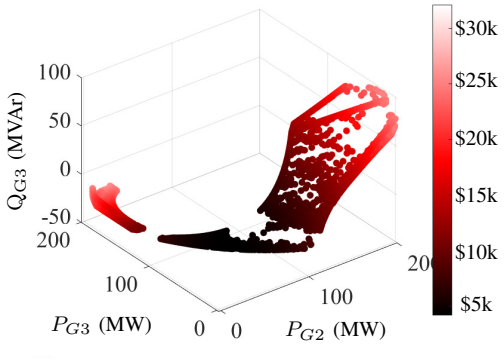


Figure 1. Points in the OPF feasible space of the acyclic three-bus test case from [20] obtained by repeatedly solving (2). Without enforcing (3), these points are not distributed evenly throughout the feasible space.

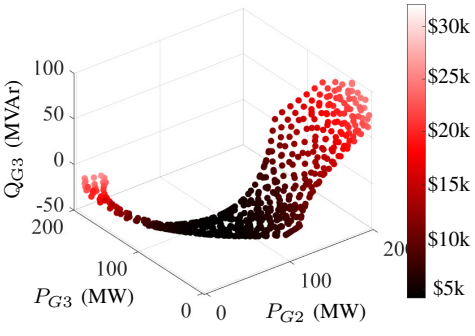


Figure 2. Points in the OPF feasible space of the acyclic three-bus test case from [20]. Augmenting (2) with (3) results in the computed points being more evenly distributed throughout the OPF feasible space.

C. Continuation Power Flow

Starting from an initial point, continuation methods track a curve defined by systems of nonlinear equations using a predictor-corrector process. In power systems, continuation power flow (CPF) methods are widely used to compute steady-state stability limits ("nose curves") [15]. CPF introduces a parameterized representation of the power flow equations:

$$g(\theta, V) = \begin{bmatrix} P(\theta, V) - P^{inj} \\ Q(\theta, V) - Q^{inj} \end{bmatrix} = 0, \tag{4a}$$

$$f(\theta, V, \lambda) = g(\theta, V) - \lambda b = 0, \tag{4b}$$

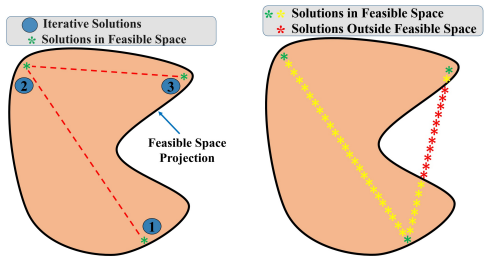
where b captures changes in loading or generation:

$$b = \begin{bmatrix} P_{target}^{inj} - P_{base}^{inj} \\ Q_{target}^{inj} - Q_{base}^{inj} \end{bmatrix}. \tag{5}$$

where "base" and "target" refer to a pair of operating points computed via a nonlinear programming solver as discussed in the previous section. The base and target points define the endpoints of a path traced out by the continuation power flow algorithm. Each solution (θ, V, λ) along the curve is parameterized to uniquely identify subsequent or preceding points [28]. In MATPOWER, a predictor estimates the next solution using the tangent vector $z^j = [d\theta, \quad dV, \quad d\lambda]_j^T$ at $((\theta, V)^j, \lambda^j)$, obtained from:

$$\begin{bmatrix} f_{(\theta,V)} & f_\lambda \\ P_{(\theta,V)}^{j-1} & P_\lambda^{j-1} \end{bmatrix} z^j = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{6}$$

where $f_{\theta,V}$ and f_λ are the partial derivatives of f with respect to θ, V and λ. This extended Jacobian includes the additional



(a) (b)

Figure 3. The proposed algorithm first finds a limited number of feasible points within the problem's feasible space by first solving (1) and then repeatedly solving (2) augmented with (3). The resulting set of feasible points is indicated by green stars in Fig. 3a. Next, the algorithm runs continuation power flows between each pair of computed points to seek additional feasible points. The red and yellow stars in Fig. 3b represent points obtained through continuation power flow. Note that some of these points shown as red stars may not be feasible for the original problem. Thus, the algorithm filters points from the continuation power flow based on their satisfaction of the OPF constraints (1b)–(1l).

column $f_\lambda = -b$ and a normalization row ensuring non-singularity. The tangent vector is normalized as

$$\bar{z}^j = \frac{z^j}{\|z^j\|_2}, \tag{7}$$

and used to predict the next point:

$$\begin{bmatrix} (\hat{\theta}, \hat{V})^{j+1} \\ \hat{\lambda}^{j+1} \end{bmatrix} = \begin{bmatrix} (\theta, V)^j \\ \lambda^j \end{bmatrix} + \beta^j \bar{z}^j, \tag{8}$$

where β^j is the continuation step size. The corrector phase refines this prediction by using Newton's method to solve the following augmented system of equations:

$$P^j(\theta, V, \lambda) = \lambda - \lambda^j - \beta^j = 0, \tag{9}$$

$$\begin{bmatrix} f(\theta, V, \lambda) \\ P^j(\theta, V, \lambda) \end{bmatrix} = 0, \tag{10}$$

corresponding to the "natural parameterization," where λ directly serves as the continuation parameter.

D. Leveraging CPF to Find Feasible Points

Our proposed algorithm begins by obtaining an initial feasible operating point by applying a nonlinear programming solver to the OPF problem (1). This solution serves as a seed for constructing the feasible set. We then calculate additional feasible points by solving (2) augmented with (3), which enforces a minimum spacing of Δ to obtain a broad coverage of the feasible space. Once this initial set of points is obtained, continuation power flow (CPF) is applied between all pairs of known feasible points to interpolate additional points that may be feasible for the OPF problem. A final filtering step removes any points violating OPF constraints such as limits on generator outputs, voltage mganitudes, or line flows. The overall workflow is illustrated in Fig. 3.

A key advantage of this algorithm is its computational efficiency. Repeatedly solving (2) with distance constraints (3) rapidly becomes impractical as the number of known points grows since each iteration involves constraints from all previous points. In contrast, the proposed combination of nonlinear programming solvers, CPF traces, and filtering efficiently explores the feasible spaces, producing well spaced samples of feasible spaces for test systems that would otherwise be computationally challenging. Algorithm 1 outlines the main steps of the proposed feasible-space sampling method.

Algorithm 1: Continuation Power Flow

Initialize parameters, system variables, spacing threshold $\Delta > 0$, and CPF step size $\Delta\lambda$.
Compute Initial Points Using a Nonlinear Programming Solver:
Solve OPF once with a nonlinear programming solver to obtain a feasible point $(\hat{P}, \hat{Q})^{(0)}$.
Set $\mathcal{F} \leftarrow \{(\hat{P}, \hat{Q})^{(0)}\}$.
Repeat until a stopping rule is met:
    Solve the modified OPF problem in (2) with (3).
    Update $\mathcal{F} \leftarrow \mathcal{F} \cup \{(\hat{P}, \hat{Q})\}$.
Continuation Power Flow Between Pairs of Points:
Enumerate all unordered pairs $\{(\hat{P}, \hat{Q})^a, (\hat{P}, \hat{Q})^b\} \subset \mathcal{F}$ with $a < b$.
For each pair, run CPF from $(\hat{P}, \hat{Q})^a$ toward $(\hat{P}, \hat{Q})^b$ with predictor–corrector steps.
For each accepted point $(\hat{P}, \hat{Q})^{(j)}$:
    Update $\mathcal{F} \leftarrow \mathcal{F} \cup \{(\hat{P}, \hat{Q})^{(j)}\}$.

IV. Feasible Space Visualizations

This section shows results from applying the proposed algorithm to selected test cases from the PGLib-OPF v18.08 benchmark library [26] and from modified IEEE test cases from [20]. These test cases are characterized by large optimality gaps between the best-known local objective values and the lower bounds obtained from convex relaxations, making them computationally challenging. Consequently, they provide a valuable benchmark for evaluating the performance, scalability, and practical value of the proposed algorithm.

The first stage of the proposed algorithm, which identifies feasible points using a nonlinear programming solver, is implemented in Julia 1.0.2 with JuMP v1.15.0 [29], PowerModels.jl [30], and Gurobi 10.0.2. The second stage, which applies the continuation power flow (CPF) technique to interpolate feasible operating points, is implemented in MATLAB using MATPOWER. Computations are performed on a laptop with an Intel i7 1.80 GHz processor and 16 GB of RAM.

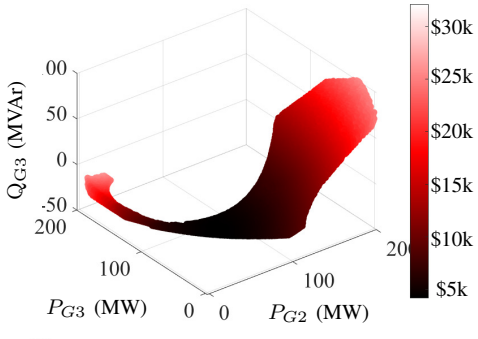


Figure 4. The feasible space for the acyclic three-bus test case from [20].

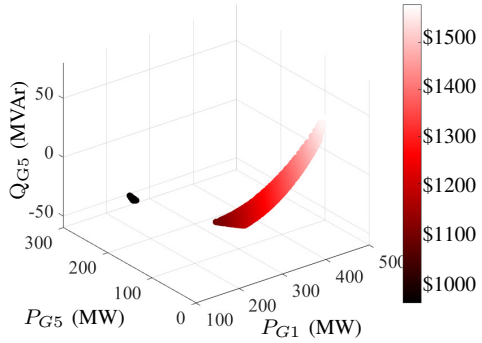


Figure 5. A projection of the feasible space for five-bus test case from [1]. This feasible space is non-convex and disconnected due to the lower limit on reactive power generation at bus 5.

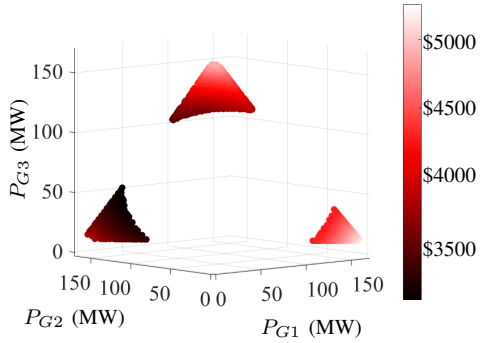


Figure 6. Feasible Space for the nine-bus system from [1].

The choice of the parameter Δ plays an important role in the performance of the proposed algorithm. We select Δ based on the range of generator active and reactive power limits, $[\underline{P}_i^g, \overline{P}_i^g]$ and $[\underline{Q}_i^g, \overline{Q}_i^g]$, as well as overall size of the system. Larger values of Δ are used for wider feasible intervals to allow more effective exploration of the feasible space. Note that a Δ value that works well for one test case may slow down another, so some trial-and-error tuning is needed to balance exploration and execution time.

Fig. 4 shows the feasible space of the three-bus test case from [20]. In this case, the proposed algorithm identified approximately 400 feasible points within two iterations of the first stage. The second stage, which executes CPF between the computed points, completed in 46 minutes. Some CPF iterations required significantly more computation time than others due to slow convergence near feasibility boundaries. To address this, we introduced a time threshold. If any CPF iteration exceeded a predefined time limit, the CPF process for that pair of points was skipped. This adjustment reduced the total computation time by approximately 15% without substantively affecting the completeness of the feasible space.

The most notable result for this case is the execution time comparison. While the algorithm in [17] required several hours of computation on a high-performance computing cluster to obtain thorough samplings of the feasible spaces for different OPF test cases, the proposed algorithm achieved a comparable result in less than one hour on a single laptop. Although the algorithm in [17] guarantees complete feasible-space coverage to within a specified discretization tolerance, visually comparing the feasible spaces obtained by both methods (as shown in Fig. 4) shows that the proposed approach accurately reconstructs the entire feasible space at a fraction of the computational cost. This result highlights the proposed algorithm's ability to efficiently capture the OPF feasible space.

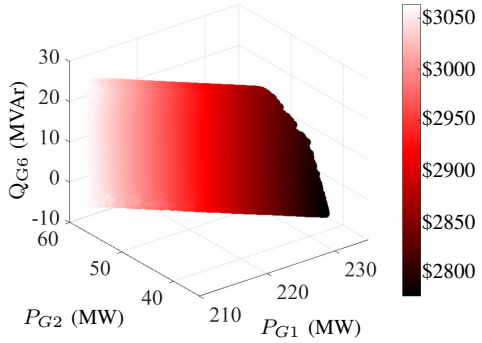


Figure 7. Feasible space of case14_ieee_sad from [26].

A similar observation can be made for the "WB5" test case, whose feasible space is shown in Fig. 5. The feasible space computed by the proposed algorithm matches that obtained using the algorithm in [17], which guarantees the full feasible-space coverage. However, the computational efficiency of the proposed method is significantly higher. The approach in [17] required 3.65 hours on the Fusion cluster at the Argonne National Laboratory, using several nodes that each have eight-core 2.6 GHz Xeon processors and 32 GB of memory. In contrast, the proposed algorithm produced a similarly detailed sampling of the feasible space in about 100 minutes on a laptop, demonstrating a substantial computational speedup.

The performance gain is most evident in the "case9mod" system. As shown in Fig. 6, the proposed algorithm reproduces a visually similar sampling of the feasible space as [17] but reduces computation time from 254.3 hours on a cluster to about 22 hours on a laptop, over an order of magnitude faster. This confirms its high accuracy, scalability, and efficiency. For the larger "case14_ieee_sad" and "case30_ieee" systems, the proposed algorithm again demonstrates its scalability. As shown in Fig. 7, the algorithm successfully computed samples of the feasible space for "case14_ieee_sad", previously intractable for [17], in roughly 32 hours. Similarly, the proposed algorithm was applied to the "case30_ieee" test case, whose feasible space is shown in Fig. 8. This test case represents an even larger system, further challenging the scalability of feasible-space computation methods. The algorithm in [17] was not able to compute the feasible space for this case due to the same computational limitations encountered in "case14_ieee_sad." However, the proposed algorithm computed the feasible space of "case30_ieee" in 45 hours. Again, this test case was intractable using the algorithm in [17].

These results demonstrate that the proposed approach effectively samples feasible spaces for small and medium OPF problems, scaling to systems that were previously intractable (e.g., "case14_ieee_sad" and "case30_ieee"). The algorithm's flexibility also enables generation of less-dense feasible spaces in much shorter runtimes, offering a tunable trade-off between computational effort and resolution for various power system analysis and planning applications. Across all test cases, the results show that the proposed algorithm has execution

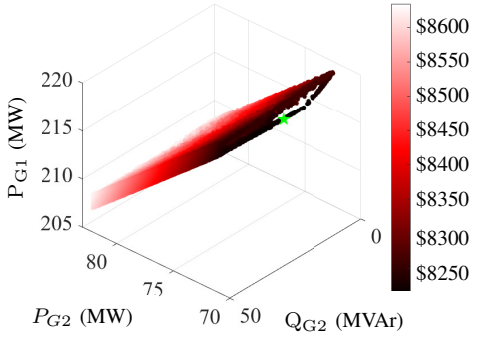


Figure 8. Feasible space of case30_ieee from [26].

times that are one to two orders of magnitude faster than existing methods. The figures confirm that the feasible spaces match those in [17], validating the results' accuracy. Overall, combining nonlinear programming solver based exploration with continuation power flow interpolation yields a scalable, accurate, and computationally efficient algorithm for sampling and visualizing OPF feasible spaces.

V. Conclusion

This paper has presented an algorithm for computing points within the feasible spaces of OPF problems. To achieve this, the algorithm first solves the OPF problem to obtain a feasible point. The algorithm then repeatedly solves a modified OPF problem that maximizes the distance between the newly computed point and the previously found feasible points. By enforcing an additional constraint in the modified OPF problem, the algorithm avoids finding feasible points that are too close to each other to spread out the points across the feasible space. Subsequently, the proposed algorithm employs a continuation power flow to identify more feasible points between each pair of previously computed points. The algorithm is able to sample and visualize the feasible spaces of challenging OPF problems. Future research includes using the proposed algorithm to study other OPF test cases to gain a deeper understanding of the physical characteristics associated with challenging OPF problems.

References

[1] W. A. Bukhsh, A. Grothey, K. I. McKinnon, and P. A. Trodden, "Local solutions of the optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4780–4788, 2013.

[2] D. Bienstock and A. Verma, "Strong NP-hardness of AC power flows feasibility," *Operations Research Letters*, vol. 47, no. 6, pp. 494–501, 2019.

[3] J. L. Carpentier, "Contribution a l'etude du dispatching economique," *Bulletin de la Societe Francoise des Electriciens*, vol. 8, no. 3, pp. 431–447, 1962.

[4] J. Momoh, R. Adapa, and M. El-Hawary, "A review of selected optimal power flow literature to 1993. Parts I and II," *IEEE Transactions on Power Systems*, vol. 14, no. 1, pp. 96–111, Feb. 1999.

[5] F. Capitanescu, "Critical review of recent advances and further developments needed in AC optimal power flow," *Electric Power Systems Research*, vol. 136, pp. 57–68, 2016.

[6] K. Lehmann, A. Grastien, and P. Van Hentenryck, "AC-feasibility on tree networks is NP-hard," *IEEE Transactions on Power Systems*, vol. 31, no. 1, pp. 798–801, 2015.

[7] I. A. Hiskens and R. J. Davy, "Exploring the power flow solution space boundary," *IEEE Transactions on Power Systems*, vol. 16, no. 3, pp. 389–395, 2001.

[8] K. Dvijotham and K. Turitsyn, "Construction of power flow feasibility sets," *arXiv:1506.07191*, 2015.

[9] Y. V. Makarov, Z. Y. Dong, and D. J. Hill, "On convexity of power flow feasibility boundary," *IEEE Transactions on Power Systems*, vol. 23, no. 2, pp. 811–813, 2008.

[10] J. Lavaei, D. Tse, and B. Zhang, "Geometry of power flows and optimization in distribution networks," *IEEE Transactions on Power Systems*, vol. 29, no. 2, pp. 572–583, 2013.

[11] M. R. Narimani, K. R. Davis, and D. K. Molzahn, "Certifying the nonexistence of feasible path between power system operating points," *arXiv:2509.05935*, 2025.

[12] D. A. Alves and G. Da Costa, "An analytical solution to the optimal power flow," *IEEE Power Engineering Review*, pp. 49–51, 2002.

[13] B. C. Lesieutre and I. A. Hiskens, "Convexity of the set of feasible injections and revenue adequacy in FTR markets," *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 1790–1798, 2005.

[14] H.-D. Chiang, I. Dobson, R. J. Thomas, J. S. Thorp, and L. Fekih-Ahmed, "On voltage collapse in electric power systems," *IEEE Transactions on Power systems*, vol. 5, no. 2, pp. 601–611, 1990.

[15] V. Ajjarapu and C. Christy, "The continuation power flow: A tool for steady state voltage stability analysis," *IEEE Transactions on Power Systems*, vol. 7, no. 1, pp. 416–423, 1992.

[16] F. Alvarado, I. Dobson, and Y. Hu, "Computation of closest bifurcations in power systems," *IEEE Transactions on Power Systems*, vol. 9, no. 2, pp. 918–928, 1994.

[17] D. K. Molzahn, "Computing the feasible spaces of optimal power flow problems," *IEEE Transactions on Power Systems*, vol. 32, no. 6, pp. 4752–4763, 2017.

[18] A. Venzke, D. Molzahn, and S. Chatzivasileiadis, "Efficient creation of datasets for data-driven power system applications," *Electric Power Systems Research*, vol. 190, January 2021, presented at *21st Power Systems Computation Conference (PSCC), June 2020.*

[19] D. Mehta, H. D. Nguyen, and K. Turitsyn, "Numerical polynomial homotopy continuation method to locate all the power flow solutions," *IET Generation, Transmission & Distribution*, vol. 10, no. 12, pp. 2972–2980, 2016.

[20] M. R. Narimani, D. K. Molzahn, D. Wu, and M. L. Crow, "Empirical investigation of non-convexities in optimal power flow problems," in *American Control Conference (ACC)*, 2018, pp. 3847–3854.

[21] I. V. Nadal and S. Chevalier, "Optimization-based exploration of the feasible power flow space for rapid data collection," in *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 2022, pp. 347–352.

[22] F. Thams, A. Venzke, R. Eriksson, and S. Chatzivasileiadis, "Efficient database generation for data-driven security assessment of power systems," *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 30–41, 2019.

[23] T. Joswig-Jones, K. Baker, and A. S. Zamzam, "OPF-Learn: An open-source framework for creating representative AC optimal power flow datasets," in *IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, 2022.

[24] C.-Y. Xue and H.-D. Chiang, "A trajectory-unified method for constructing the feasible region of OPF problems," *Electric Power Components and Systems*, vol. 48, no. 4-5, pp. 423–435, 2020.

[25] H.-D. Chiang and C.-Y. Jiang, "Feasible region of optimal power flow: Characterization and applications," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 236–244, 2017.

[26] IEEE PES Task Force on Benchmarks for Validation of Emerging Power System Algorithms, "The Power Grid Library for Benchmarking AC Optimal Power Flow Algorithms," *arXiv:1908.02788*, Aug. 2019. [Online]. Available: https://github.com/power-grid-lib/pglib-opf

[27] R. Zimmerman, C. Murillo-Sánchez, and R. Thomas, "MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education," *IEEE Transactions on Power Systems*, no. 99, pp. 1–8, 2011.

[28] H.-D. Chiang, A. J. Flueck, K. S. Shah, and N. Balu, "CPFLOW: A practical tool for tracing power system steady-state stationary behavior due to load and generation variations," *IEEE Transactions on Power Systems*, vol. 10, no. 2, pp. 623–634, 1995.

[29] I. Dunning, J. Huchette, and M. Lubin, "JuMP: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 259–320, June 2017.

[30] C. Coffrin, R. Bent, K. Sundar, Y. Ng, and M. Lubin, "PowerModels.jl: An Open-Source Framework for Exploring Power Flow Formulations," in *Power Systems Computation Conference (PSCC)*, June 2018.