A Robust Method for Applying Forecast States to AC Power Flow Cases

Thomas Nicol
PowerWorld Corporation
Champaign, IL, USA
thomas.nicol@powerworld.com

Thomas J. Overbye
Electrical and Computer Engineering
Texas A&M University, College Station, TX, USA
overbye@tamu.edu

***Abstract*—Power system studies frequently require the application of forecast data, such as load or renewable generation projections, to an existing solved power flow case. However, direct application of forecast values may result in convergence failure due to the nonlinear nature of the AC power flow equations. This paper presents a simple and robust algorithm for applying arbitrary forecast data while maintaining solvability. The method parameterizes forecast application as a scalar interpolation between a reference state and a target forecast state and iteratively searches for solvable intermediate operating points using adaptive step reduction. When a solvable intermediate state is identified, it is used as a new reference, allowing the algorithm to progressively advance toward the full forecast condition. The approach requires no modification to the underlying power flow solver and operates as a wrapper around existing tools. The method is evaluated on a 10,000-bus synthetic system across 168 hourly scenarios with varying solar availability and load forecasts. Results demonstrate that the algorithm reliably recovers solvable cases that fail under direct forecast application, with modest computational overhead. The impact of step size parameters on robustness and performance is also analyzed.**

***Index Terms*—Power flow analysis, continuation methods, load forecasting, numerical methods, power system simulation, convergence analysis, voltage stability, nonlinear systems, transmission system analysis, scenario analysis.**

I. Introduction

Power system planning and operational studies frequently rely on forecast data to evaluate anticipated system conditions. Load projections, weather forecasts, interchange schedules, etc., all require the modification of a base power flow state with specific projected conditions; the wider impact on the rest of the system state must be determined by running a power flow solution with these modifications in place. However, direct application of forecast data to an existing case may not produce a solvable AC power flow state. Large changes in system parameters can render the case unable to converge due to any number of nonlinear effects.

One practical approach is to apply forecast changes incrementally, allowing the power flow solver to converge at intermediate states, gradually transitioning the system from the reference condition to the forecast condition. This is conceptually related to continuation techniques [1] used in voltage stability analysis, where system loading is gradually varied while maintaining the ability to solve. However, traditional continuation power flow methods typically require modification of the solver algorithm and make small incremental steps based on the Jacobian.

This paper presents a simple and robust algorithm for transitioning a power flow case from one solved state to another with the application of arbitrary forecast data. The proposed method treats forecast application as a parameterized transition between the reference state and the forecast state, scaling the application of forecast values by a controllable application parameter. Starting from a solved reference state, the algorithm attempts to apply the full forecast and solve the resulting power flow. If the solve fails, the forecast application level is reduced and the solve is retried. Once a solvable intermediate state is found that state becomes the new reference point from which further forecast application is attempted. Through this iterative process, the algorithm progressively advances the system toward the target forecast state by finding intermediate states which also solve.

The primary advantages of the proposed approach are that it adaptively identifies the largest solvable step at each iteration, and operates as a wrapper around an existing power flow solver requiring no special modification to the underlying solution algorithm to accommodate specific types of forecast modifications being applied. This makes the method applicable in a wide range of analysis environments including commercial power flow tools and automated study workflows and enables automated application of arbitrary forecast data. The effectiveness of the approach is demonstrated using a synthetic large-scale power system case (ACTIVSg10k, publicly available in the data repository described in [2]) with forecast modifications to load [3] and available solar power. Results show that the algorithm can reliably obtain solvable cases that would otherwise fail when forecast changes are applied directly.

II. Forecast Scaling Formulation

To enable algorithmic application of forecast data, the transition from a solved reference state to a forecast state can be expressed as a parameterized interpolation between the two operating points.

A. Forecast Parameterization

Let P_{ref} represent the vector of reference values for arbitrary parameters in the solved reference state and let P_{for} represent the vector of corresponding forecast values. A scalar parameter $\alpha \in [0,1]$ is introduced to represent the fraction of the forecast that has been applied. The parameter value applied to the case at each iteration is then defined as

$$P(\alpha) = P_{ref} + \alpha\left(P_{for} - P_{ref}\right)$$

Under this formulation, $\alpha = 0$ corresponds to the solved reference state, while $\alpha = 1$ represents the full forecast

This material is based upon work supported by the U.S. Department of Energy's Office of Cybersecurity, Energy Security, and Emergency Response under the Energy Delivery and Security Directorate award Number DE-CR0000067. The views expressed herein do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

scenario. Intermediate values of α represent partial application of the forecast, linearly scaling the deviation between the reference and forecast values. The parameterization can be applied to any modeled quantity that is modified by the forecast, including generator outputs, load demand, interchange targets, limits introduced by variable weather conditions, or any other state change required by some arbitrary type of forecast. This approach provides a unified mechanism for representing a wide range of forecast-driven modifications within a single scalar parameter.

B. Application to Power Flow

Let x denote the vector of system state variables (e.g. bus voltage magnitudes and phase angles). The AC power flow equations can then be written as a parameterized nonlinear system.

$$F(x, \alpha) = 0$$

where the function $F(\cdot)$ represents the standard power balance equations with modifications determined by the forecast scaling relationship above. The objective of the forecast application process is to move from the known solution $(x_0, \alpha = 0)$ to a solution at $\alpha = 1$ with the forecast values fully applied in a solved state.

In practice, however, the nonlinear nature of the power flow equations can make convergence difficult when large parameter changes are introduced in a single step. Even when both the reference case and the forecast case individually represent reasonable operating conditions, the intermediate states encountered during the transition may cause the power flow solver to diverge or violate operational limits. Directly solving the system with $\alpha = 1$ may therefore fail even when a nearby solvable state exists.

The goal of the proposed approach is to search the intermediate space for solvable states that can iteratively nudge the underlying system data into a state from which a solution at full application can be obtained.

III. Iterative Forecast Application Algorithm

A. Algorithm Overview

To efficiently navigate the space from the reference state $\alpha = 0$ to the forecast state $\alpha = 1$, we perform an adaptive binary search along the forecast application parameter α using the success or failure of the power flow solution as the primary feedback signal.

The algorithm begins by attempting to apply the full forecast. If the resulting power flow converges, the process terminates with a solved case. If the solve attempt fails, the algorithm cuts the forecast application level in half and retries. Once a solvable intermediate state is found that state becomes the new reference point, and the process repeats to advance further toward the target forecast. Through successive iterations, the method leapfrogs through a series of intermediate solutions that monotonically approach $\alpha = 1$.

This approach can be interpreted as a continuation-like method with adaptive step sizing, where the step size is determined implicitly through a backtracking based on solver convergence behavior.

B. Algorithm Steps

Let $\alpha_{target} = 1.0$ represent the full forecast application level. The algorithm maintains a current reference state corresponding to a solved operating point at some α_{ref}. At each stage, the goal is to find a new solvable state at a higher value of α. To prevent the indefinite asymptotic approach of a truly insoluble state, we also set some minimum step size Δ_{min} and declare failure to solve if $\alpha - \alpha_{ref} < \Delta_{min}$.

The procedure is as follows:

1) Initialization

Solve the base power flow case to obtain a reference state x_{ref} at $\alpha_{ref} = 0$. Cache all quantities P_{ref} that will be modified by the forecast.

2) Attempt Full Forecast Application

Set $\alpha = \alpha_{target}$ and apply the full forecast values. Attempt to solve the power flow. If it solves, success. If not, continue.

3) Backtrack

If the power flow does not converge at α_{target}, reduce the application level by half toward the reference state.

$$\alpha \leftarrow \frac{\alpha + \alpha_{ref}}{2}$$

Reapply the forecast at the updated α and attempt to solve again. This process is repeated until either:

a) A solvable intermediate state is found, or

b) $\alpha - \alpha_{ref} < \Delta_{min}$. and the attempt is abandoned.

4) Accept Intermediate Solution

If a solution is obtained at some $\alpha < \alpha_{target}$, update the reference state.

$$x_{ref} \leftarrow x, \alpha_{ref} \leftarrow \alpha$$

This establishes a new baseline from which further forecast application can be attempted.

5) Iterate Towards Target

Repeat steps 2-4 until either:

a) $\alpha_{ref} = \alpha_{target}$ with a full application of the forecast values, or

b) The algorithm terminates due to Δ_{min}

C. Algorithm Properties

The proposed method has several important characteristics:

a) Monotonic Progression

Each accepted solution increases α_{ref}, ensuring that the algorithm makes forward progress toward the forecast state

b) Adaptive Step Size

The effective step size is determined dynamically through backtracking. Larger steps are taken when the system is well-behaved, while smaller steps are used in regions where convergence is more difficult.

c) Binary Search Behavior

The backtracking process performs a bisection-like search along the interval $[\alpha_{ref}, \alpha_{target}]$, identifying the largest solvable step at each iteration.

d) Solver-Agnostic Implementation

The method requires only the ability to apply modified values and detect convergence, making it compatible with existing power flow solvers without requiring solver modification or any consideration of how arbitrary forecast values interact with the underlying solution process.

e) Piecewise Continuation

By resetting the reference state after each successful step, the algorithm effectively performs continuation across a sequence of locally linear regions, improving robustness for large forecast deviations.

D. Termination Criteria

The algorithm terminates successfully when $\alpha_{ref} = \alpha_{target}$ indicating the full forecast values have been applied successfully, or when $\alpha - \alpha_{ref} < \Delta_{min}$ indicating that we are applying a step size smaller than our predetermined threshold. These parameters provide a mechanism to balance computational effort against robustness.

IV. Experimental Setup

The proposed algorithm was evaluated using a large-scale synthetic transmission network consisting of approximately 10,000 buses. The study was designed to assess the algorithm's ability to reliably transition solved base cases to forecast conditions across a range of operating scenarios and parameter settings.

A. Study Cases

A set of 168 cases were configured to represent hourly operating conditions over the course of one week. Each case was initialized from a common base network model and then customized to reflect time-varying system conditions in the form of solar generation availability. Historical cloud-cover data was applied to each instance of ACTIVSg10k, calculating max MW values for solar resources throughout the system. This produced realistic temporal variability in renewable generation, including periods of reduced output corresponding to increased cloud cover or nightfall.

Following this preprocessing step, each case was solved to establish a valid reference operating point. These solved cases served as the starting points for the application of additional forecast data using the proposed algorithm.

B. Forecast Application

Synthetic load forecast data was applied to each case to represent anticipated demand conditions. The forecast consisted of modifications to aggregate load by Area, with no corresponding generation forecasts specified. As a result, the burden of balancing generation with the modified load was handled implicitly by the power flow solver.

All simulations were performed using PowerWorld Simulator [4], set to control generation by AGC with minimal configuration. Other control mechanisms (e.g. Economic Dispatch) could also be configured but were not used in this study. The solution itself was run using a standard Newton-Raphson approach.

The forecast data was applied iteratively, scaling Area load at each step to values determined by $P(\alpha)$, with the algorithm searching intermediate states until the full forecast was achieved or termination criteria was met.

C. Parameter Variations

To evaluate the impact of algorithm parameters on performance, three independent runs were conducted using different termination thresholds:

- Low depth $\Delta_{min} = 0.05$
- Medium depth $\Delta_{min} = 0.01$
- High depth $\Delta_{min} = 0.0025$

The minimum step size defines the smallest allowable increment in forecast application during the backtracking process. If the algorithm is unable to identify a solvable state within this threshold, the case is classified as a failure. Smaller values of Δ_{min} allow more fine-grained exploration of difficult points in the parameter space, potentially improving robustness at the cost of additional power flow solution attempts.

Each of the 168 cases was processed independently for all three parameter settings, resulting in a total of 504 algorithm executions.

D. Evaluation Metrics

The performance of the algorithm was evaluated using the following metrics:

a) Success Rate

The proportion of cases for which the algorithm successfully achieved a solved state at $\alpha_{ref} = \alpha_{target}$

b) Iteration Count

The total number of forecast application attempts required to reach a termination condition, including both successful and unsuccessful solves.

c) Run Time

The total time taken to run each set of cases using 18 parallel instances of Simulator under the management of PowerWorld Cruncher [5].

These metrics were used to compare the effectiveness and computational cost of different minimum step size settings, as well as to characterize the algorithm's behavior across a diverse set of operating conditions.

V. Results

The performance of the proposed algorithm was evaluated across 168 hourly cases for three minimum step size settings, corresponding to Low, Medium, and High computational effort. The results demonstrate the tradeoff between robustness and computational cost, as well as the characteristic convergence behavior of the algorithm.

A. Overall Performance

In each run, 40 of the 168 cases solved on the initial forecast application, and the remaining 128 required some level of iterative application to achieve a solved state with full forecast values applied. Of those, 32 only required a single intermediate solved state $\alpha = 0.5$ to successfully make the jump to full forecast application.

For the remaining cases, the algorithm exhibited the expected binary search behavior, progressively identifying solvable intermediate values of α through step halving. The number of iterations attempted varied depending on system conditions and the minimum step size constraint.

Table 1 summarizes the aggregate performance across the three parameter settings.

TABLE 1
Overview of Evaluation Metrics across runs

Δ_{min}	**Failures**	**Iterations**	**Runtime**
0.05 (L)	17/168	846	15:58
0.01 (M)	4/168	949	16:46
0.0025 (H)	3/168	980	18:54

Reducing the minimum step size significantly improved the success rate. The number of failed cases decreased from 17 in the Low-depth configuration to 4 in the Medium-depth configuration, with only a marginal additional improvement to 3 failures in the high-effort configuration. This suggests diminishing returns as Δ_{min} is reduced beyond a certain threshold.

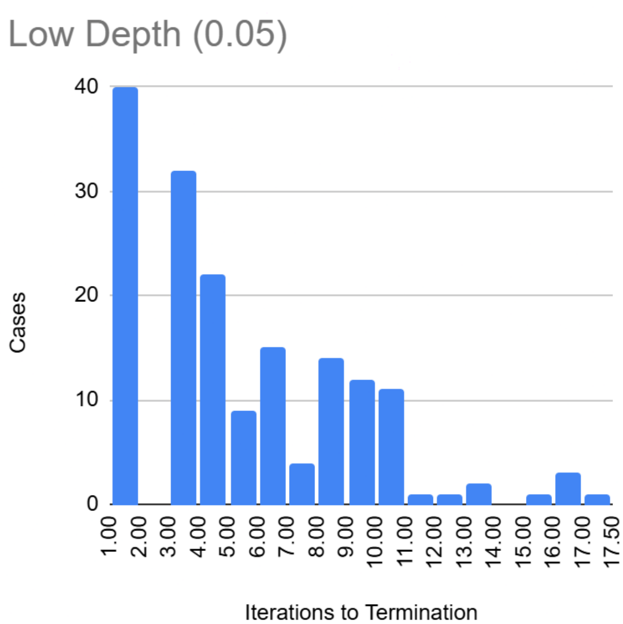

Fig. 1. Distribution of iteration counts required to reach termination for the Low-depth configuration min=0.05. Most cases converge within a small number of iterations, with a limited number of difficult cases requiring additional backtracking.

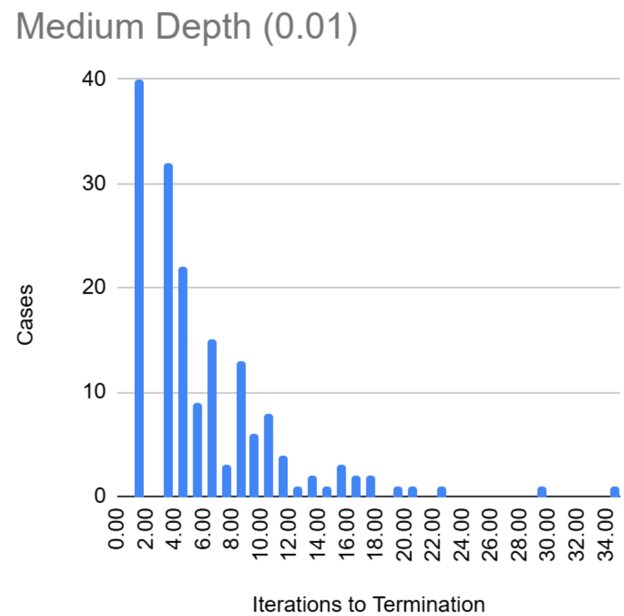

Fig. 2. Distribution of iteration counts for the Medium-depth configuration min=0.01. Compared to the Low-depth case, the distribution extends to higher iteration counts as the algorithm explores finer-grained intermediate states.

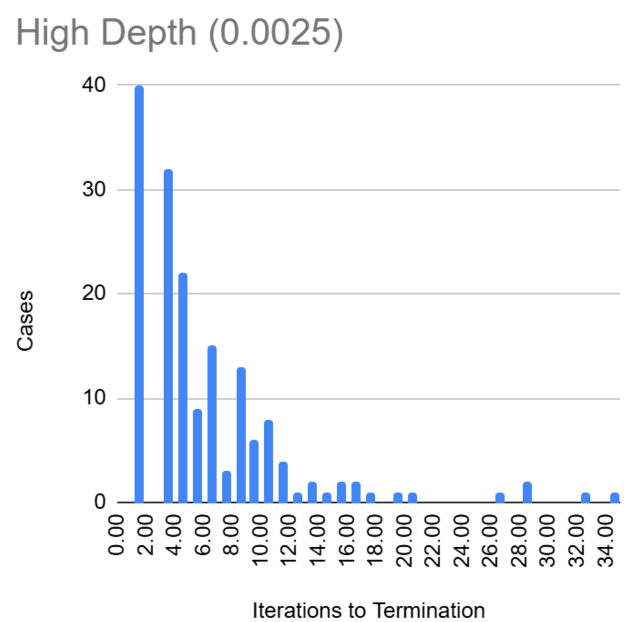

Fig. 3. Distribution of iteration counts for the High-depth configuration min=0.0025. The increased resolution allows deeper exploration of the parameter space, resulting in higher iteration counts for challenging cases while maintaining similar behavior for most scenarios.

The increase in computational effort was modest overall and concentrated on the few problem cases. Total iterations increased by approximately 16% from the Low-depth to High-depth configurations, while total runtime increased by approximately 18%. This indicates that finer step resolution improves robustness without a disproportionate increase in computational cost when most cases achieve full forecast application relatively quickly.

B. Iteration Distribution

Histograms of iteration counts for each configuration provide additional insight into algorithm performance. In all cases, most scenarios required only a small number of iterations, with a long tail of more difficult cases requiring additional backtracking steps.

As Δ_{min} decreased, the distribution allowed for higher iteration counts as the algorithm searched deeper for intermediate solution states.

C. Case Study: Difficult Scenario

A representative difficult case was examined in detail to explore the behavior of the algorithm under challenging conditions. This case failed to reach full forecast solution in the Low-depth and Medium-depth configurations but successfully converged in the High-depth configuration.

Figure 4 shows the sequence of attempted α values during the search process for this case, with green circles representing solved reference states, and red Xs representing failed solutions. Blue stars indicate solved reference states at the farthest progression made by the Low-depth $\left(\alpha_{ref} = 0.6309\right)$ and Medium-depth $\left(\alpha_{ref} = 0.9849\right)$ attempts respectively, each terminating before finding a fully applied solved state due to their minimum step-size limitations.

This example highlights the importance of step size resolution in capturing narrow regions of solvability that may not be reachable within coarser parameter increments.

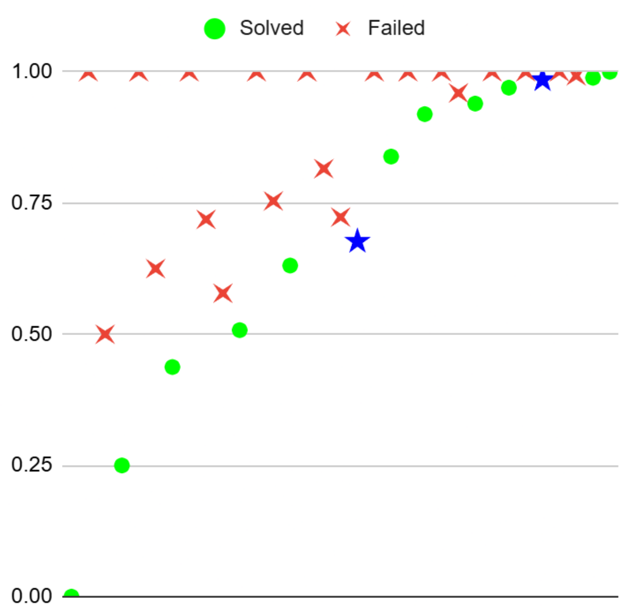

Fig. 4. Sequence of attempted forecast application values (α) for a representative difficult case over 32 iterations. Green markers indicate successful solutions, while red markers indicate failed solve attempts. Blue stars denote the maximum α_{ref} achieved in the Low-depth and Medium-depth configurations prior to termination. The High-depth configuration successfully identifies additional intermediate states, enabling convergence to the full forecast condition.

VI. Discussion

The results demonstrate that the proposed algorithm provides a robust and efficient mechanism for applying forecast data to power flow cases. Even relatively coarse step size settings can recover solvable states for most cases, while finer step sizes offer incremental improvements in robustness. This method can be interpreted as a continuation-like approach that replaces explicit tangent prediction with an adaptive, solver-driven search for feasible points along the forecast parameter axis. Unlike traditional continuation power flow methods, the proposed approach requires no modification to the underlying solver and can be implemented entirely as an external wrapper.

The presence of diminishing returns suggests that moderate step size settings (e.g. Δ_{min}= 0.01) may provide a practical balance between computational effort and success rate for large-scale studies, although the overall trade-off of a smaller Δ_{min} is minimal when fine-grained iterations are required on only a few problem cases. Most solvability barriers in the forecast parameter space appear to be relatively coarse, with only a small number of cases requiring fine-grained exploration near the boundary of feasibility.

Overall, the algorithm consistently identifies solvable intermediate states and successfully advances the system toward the forecast condition across a variable range of operating scenarios.

VII. References

[1] 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, Feb. 1992.

[2] S. Kunkolienkar, F. Safdarian, J. Snodgrass, A. Birchfield, T. Overbye, "A Description of the Texas A&M University Electric Grid Test Case Repository for Power System Studies", 2024 IEEE Texas Power and Energy Conference (TPEC), College Station, TX, Feb. 2024.

[3] H. Li, J. H. Yeo, A. L. Bornsheuer and T. J. Overbye, "The Creation and Validation of Load Time Series for Synthetic Electric Power Systems," in IEEE Transactions on Power Systems, vol. 36, no. 2, pp. 961-969, March 2021, doi: 10.1109/TPWRS.2020.3018936.

[4] PowerWorld Corporation, "PowerWorld Simulator (Version 24)," [Software] Champaign, IL, USA. Available: https://www.powerworld.com

[5] PowerWorld Corporation, "PowerWorld Cruncher (Version 1b)," [Software] Champaign, IL, USA. Available: https://www.powerworld.com