Stochastic Look-Ahead Commitment: A Case Study in MISO

Bernard Knueven, Mohammad Faqiry, Manuel Garcia, Yonghong Chen, Roger Treinen, Trevor Werho, Junshan Zhang, Vijay Vittal, Long Zhao, Anupam Thatte, Shengfei Yin

***Abstract*—This paper introduces the Stochastic Look Ahead Commitment (SLAC) software prototyped and tested for the Midcontinent Independent System Operator (MISO) look ahead commitment process. SLAC can incorporate hundreds of wind, load and net scheduled interchange (NSI) uncertainty scenarios. It uses a progressive hedging method to solve a two-stage stochastic unit commitment. The first stage optimal commitment can cover the uncertainties within the next three hours. The second stage includes both the dispatch for each of the scenarios and the commitment decisions that can be deferred. Study results on 15 MISO production days show that SLAC may bring economic and reliability benefits under uncertainty.**

***Index Terms*—Stochastic Optimization, Unit Commitment, Uncertainty Management**

I. INTRODUCTION

With the growing levels and corresponding uncertainty and variability of stochastic resources, along with associated constantly changing operating conditions and more frequent extreme weather events, power system operators are faced with significant challenges in operating the system securely. Existing tools are based on deterministic optimization models that consider a small range of scenarios. . This paper introduces the development of a prototype advisory tool – Stochastic Look-Ahead Commitment (SLAC) – that may potentially be used by system operators to enhance system security and improve energy market surplus under growing uncertainty and variability. The tool can potentially help independent system operators and regional transmission organizations (ISO/RTOs) to enhance their modeling and management of key operational uncertainties (e.g., associated with renewables, contingencies, interchange, load, loop flow, generator non-compliance, demand response). These model and operational enhancements can ultimately lead to a more secure, more efficient market with improved market transparency, price signals, and incentives for market participants.

The amount of renewable energy resources is growing significantly within the MISO footprint, increasing the amount of uncertainty that grid operators must manage. Today, ISO/RTOs use deterministic clearing engines, and offline studies and statistical analysis of historical data for a subset of inputs. For example, headroom margin is applied in the forward Reliability Assessment Commitment (RAC) and Look Ahead Commitment (LAC) processes to allow additional capacity to handle input data uncertainties.

SLAC leverages statistical information from an ensemble of potential operational scenarios and their respective likelihood. For a given time period, the SLAC will calculate an optimal solution (e.g., commitment and schedules) over the study period that maximizes the expected market surplus, in the statistical sense, over all the operational scenarios it considers. This is unlike existing security-constrained optimal power flow solutions and security-constrained unit commitment approaches that do not explicitly model a robust set of operational uncertainties and the known variability associated with power system assets, such as stochastic resources. Rather, today's tools solve two to three cases independently, where one or two system parameters, such as forecasted load level (e.g., a low, base, and high forecast), are varied and the resulting scenarios are analyzed independently. References [1, 2] introduced how MISO operations manage uncertainties through a multi-stage commitment process (illustrated in Fig. 1), reserve products, "headroom" and multiple scenarios. LAC runs every 15 minutes with three hours look ahead, and is the last stage of the commitment process. A robust look ahead commitment was prototyped in 2013 [2] and showed potential operational benefit. However, it also indicated computational challenges.

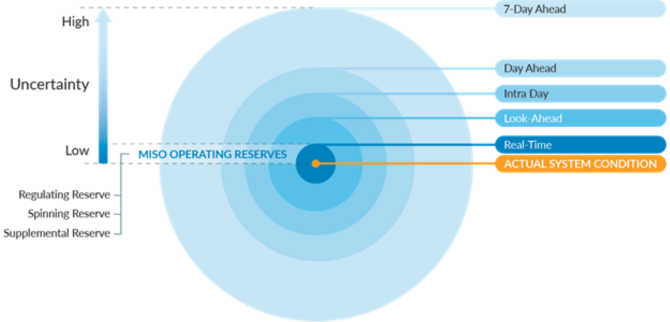

Fig. 1. Expected Level of Uncertainties throughout the multi-stage Market Clearing Processes

In the existing literature, stochastic unit commitment is typically studied in the day-ahead or longer context; see [3, 4, 5, 6] for recent overviews of both formulation and solution methodologies. Like all ISO/RTOs in the United States, MISO formulates and solves its operational problems as mixed-integer linear programs (MILP) [7]. The proposed methodology in the present work combines these MILP formulations with a customized progressive hedging algorithm (PH) [8], a classical dual-decomposition scheme for stochastic optimization. PH has been applied with success on small- to medium-scale systems for day-ahead or longer unit commitment considering load and/or renewables uncertainty [9, 10, 11, 12, 13].

Unlike most stochastic unit commitment problems studied in the literature, a typical SLAC problem has relatively few committable resources available (most units are committed by the day-ahead market), study a shorter time-horizon with higher fidelity (three hours with 15-minute windows), and is re-solved frequently (every 15 minutes). These distinguishing features lead to some modeling choices, such as second-stage binary variables, which are not easily handled by many decomposition methods for stochastic optimization problems.

The contributions of this paper are as follows: (1) we propose a novel stochastic unit commitment formulation specific to the SLAC context, utilizing generator-specific non-anticipatory constraints; (2) we demonstrate the utility of stochastic optimization in a power system context utilizing real-world historical data from a large power-system operator (MISO); (3) we perform extensive validation with MISO to ensure the complexities of MISO's market rules and implementation requirements are reflected in the SLAC prototype tool; and (4) we show that a classical decomposition method can be applied to such stochastic optimization problems in a reliable and timely manner, i.e., demonstrating good computational performance on over a thousand SLAC problem instances, each considering scheduling power production for several hundreds of generators, commitment status of over one hundred fast-start units, while satisfying reserve and transmission constraints.

The remainder of this paper is outlined as follows. In Section II, we introduce the SLAC problem formulation. In Section III, we discuss the SLAC solution methodology and computational performance. In Section IV, we evaluate the performance of SLAC against two deterministic-equivalent problems in the context of managing grid uncertainty. In Section V we draw our conclusions and discuss future work.

II. Formulation

First consider a generic stochastic optimization problem:

$$\text{minimize } [14]\ [14]\ [15](c \cdot x) + \textstyle\sum_{s \in S} p_s(\, d_s \cdot y_s)$$
$$\text{subject to: } (x, y_s) \in Q_s, \ \ \forall s \in S$$

which minimizes some (linear) objective in expectation over every scenario s, given here-and-now decisions x, which must be *identical* for every scenario s, and wait-and-see decisions y_s, which can be *different* in each scenario s. Each scenario s may take on a unique probability $p_s > 0$, with the condition $\sum_{s \in S} p_s = 1$. In the context of SLAC, the here-and-now decisions are generator commitments in time periods that must be decided on in this SLAC run – that is, commitments in time periods which will become fixed decisions in the SLAC run 15 minutes from now due to the notification time requirements of individual generators.

A. Notification Times and Non-Anticipatory Constraints

Each individual generator has a specific notification time, which represents the amount of time they require to be notified before changing their commitment status. The SLAC fixes each generator's commitment status to its current status for the duration of the generator's notification time. In the time period immediately following the exhaustion of a generator's notification time, SLAC enforces the commitment decision in all scenarios to be identical, becoming part of the first-stage decisions x. After each generator's "first committable" time period, the commitment decisions are deemed second-stage variables (e.g., part of y_s for each scenario). These later commitment decisions can be deferred until at least the next SLAC run, when more uncertainty has been resolved. Therefore, the first-stage decisions made by the SLAC become "final," that is, because of the generator's notification time they are fixed in the very next SLAC run. This is because the notification time typically represents the time to begin the process of starting up or shutting down a generator, and this process cannot be reversed once initiated.

To summarize, as opposed to most stochastic unit commitment models, the proposed SLAC's here-and-now decisions x are determined purely by individual generators' notification time, as opposed to stages defined by fixed time periods. SLAC defers to the second stage decisions y_s all commitments which could be finalized fifteen-minutes into the future or later. In this fashion, SLAC represents the flexibility inherit in the system by not enforcing the same set of commitments across all scenarios, only those commitments which need to be finalized in the next fifteen minutes are enforced to be non-anticipative.

B. Scenario Generation

The scenario generation procedure is described in reference [15]. For MISO, this procedure develops probabilistic forecasts of the power output of wind farms in its footprint, the load from 37 local balancing authorities, and the total net-scheduled-interchange across its seams. The probability distributions capture the spatial and temporal correlations between the uncertainties from these three sources. The probabilistic forecasts are sampled 200 times and are evenly weighted. A backwards reduction method is then used based on Euclidean distance to reduce the number of scenarios to 40. This reduction method results in 40 probabilistically weighted scenarios. This process is applied, and the scenarios are updated during every 15-minute interval that the SLAC is solved.

C. LAC Operational Constraints

The operational constraints represented by the set Q_s are modeled to match MISO LAC requirements and rules pertaining to resources, transmission network, reserve zones, and overall MISO system [16]. On a resource level, Q_s includes on/off and start-up/shut-down constraints, generator limits and ramp rate constraints, minimum up/down time constraints, reserve (e.g., reg, spin, supplemental) provision constraints, and some commitment-fixing constraints to honor prior and subsequent commitment plans. On the network level, Q_s includes activated transmission and post-contingency reserve deployment constraints imposed on a predetermined set of transmission constraints fed from production [17]. The remaining constraints in Q_s relate to system-level power balance and reserve requirements.

The system-level reserve requirements in MISO LAC are satisfied through reserve constraints for regulation, contingency, and ramp capability. These constraints are implemented to hold capacity and accommodate real-time load fluctuations, contingencies, and ramp needs due to net load volatility stemming from variability and uncertainty in demand, wind generation, net scheduled interchange (NSI), and other uncertainties in the input data. The SLAC tool implements these reserve requirements in each LAC-equivalent model that is built for each uncertainty scenario. While there is the potential for a stochastic look-ahead commitment to implicitly as opposed to explicitly model some of these reserve requirements [18], all of the reserve products considered in SLAC are to address uncertainties on a time-scale faster than the 15-minute SLAC time-discretization and are also reflected in MISO's economic dispatch process. Therefore, these reserve requirements are maintained in SLAC, so the second-stage dispatch decisions reflect future SCED problems while maintaining appropriate flexibility at a sub-fifteen-minute timescale.

The LAC optimization problem with the operational constraints Q_s were modeled in *Pyomo* [19, 20, 21] in a special version of EGRET [22, 23] customized for MISO. The EGRET model was validated and benchmarked against MISO LAC by solving and comparing an extensive number of small and large test case problems including a set of 1,436 full-size LAC problems from 15 MISO operation days each containing ~1,200 generators and numerous transmission constraints. The benchmarking criteria were to obtain the same or close enough results (e.g., objective value, commitment, energy/reserve schedule, and transmission flow/violation) when comparing MISO LAC and EGRET solutions. MISO's LAC is one of the largest existing practical look-ahead unit commitment problems with complicated market rules and implementation logic. Extensive benchmarking against MISO's LAC and practical implementations of the SLAC on large complex MISO test cases is an important novelty in this study that distinguishes it from existing research applied on academic test cases.

III. Solution Methodology

A. *Progressive Hedging Algorithm*

For completeness, we present the basic PH algorithm to decompose this problem per scenario as follows:

```
1.  $k := 0$
2.  For every $s \in S$, solve
        $x_s^k := \text{argmin}_{(x,y_s)\in Q_s}\ (c \cdot x + d_s \cdot y_s)$
3.  $\bar{x}^k := \sum_{s\in S} p_s x_s^k$
4.  For every $s \in S,\ w_s^k := \rho(x_s^k - \bar{x}^k)$
5.  $k := k + 1$
6.  For every $s \in S$, solve
        $x_s^k := \text{argmin}_{(x,y_s)\in Q_s} \begin{pmatrix} c \cdot x + d_s \cdot y_s + w_s^k \cdot x \\ +\frac{\rho}{2}\|x - \bar{x}^{k-1}\|^2 \end{pmatrix}$
7.  $\bar{x}^k := \sum_{s\in S} p_s x_s^k$
8.  For every $s \in S,\ w_s^k := w_s^{k-1} + \rho(x_s^k - \bar{x}^k)$
9.  $g^k := \sum_{s\in S} p_s \|x_s^k - \bar{x}^k\|$
10. If $g^k < \epsilon$ or $k \geq \bar{k}$, STOP. Otherwise, go to Step 5.
```

The main features of the algorithm are: (1) subproblems are decomposed by scenario, so the computationally expensive steps (Step 2 and Step 6) can be done in parallel, (2) the use of the weight (Lagrangian) term w_s^k to build consensus, and (3) the regularization term ρ, which appears in both the weight-update steps (Step 4 and Step 8), and in the objective of the subproblem optimization problem (Step 6). In Steps 9 and 10 we specify that we terminate when the here-and-now decisions sufficiently agree (no more than ϵ weighted difference) or when enough ($\bar{k}$) iterations have elapsed. Additionally, we note that Step 2 gives an easily calculable lower-bound on the original problem, as the requirement that "the here-and-now decisions x must be identical for every scenario s" is relaxed to allow different here-and-now decisions in different scenarios.

When the original problem is feasible and Q_s is a convex set for every $s \in S,$ the PH algorithm provably converges in linear time [8]. In the context of SLAC, the set Q_s is *nonconvex* due to the binary nature of generator commitment status. Next, we describe a customized PH algorithm implementation used to yield a performant solver for SLAC.

B. *Progressive Hedging Implementation*

We utilize the progressive hedging implementation available as part of the *mpi-sppy* package [24], which operates on existing deterministic-equivalent Pyomo models to create equivalent stochastic optimization formulations and supporting algorithms. The PH algorithm included in mpi-sppy supports the parallel execution of subproblems in PH Step 2 and Step 6 and efficient calculation of $\bar{x}^k$ in PH Steps 3 and 6 by utilizing an MPI implementation through the Python package *mpi4py* [25]. The mpi-sppy package allows the modeler to significantly customize the PH algorithm for both performance and practical usability; those utilized for the prototype SLAC tool are described below.

1) *Subproblem Grouping or Bundling*

For the 40-scenario SLAC considered in the simulations below, we selected to group scenarios into "bundles," each with two scenarios. That is, instead of solving individual scenario subproblems in Step 2 and Step 6 of PH, we solve subproblems that each consist of two scenarios. This has several advantages: (1) the computational resources available for this study would not allow for 40-way parallelism while bundles of two still enable 20-way parallelism, (2) this allows PH to converge faster in practice, and (3) the lower-bound computable at Step 2 is empirically much stronger even with small bundles of two because each subproblem solution is not too tailored to a specific scenario [26].

2) *Tuning the Solver*

The CPLEX solver version 20.1 was used for all optimization problems in the reported results. For solving the subproblems at Step 2, we tune the CPLEX solver in the following way: we set a time limit of 30 seconds, and set the *emphasis_mip* option to 2, which instructs CPLEX to focus on getting the best achievable lower bound. For solving the subproblems at Step 6, we maintain the time limit of 30 seconds and set the *emphasis_mip* option to 1, which instructs CPLEX to focus on getting the best achievable solution. The time limits in both cases ensure we do not spend too much time on a single PH iteration. The *emphasis_mip* option at Step 2 aids in obtaining a strong initial lower bound, and the *emphasis_mip* option at Step 6 aids in obtaining new solutions as PH iterates and progresses. We also make full use of the Pyomo persistent solver interface, which enables warm-starting CPLEX at Step 6 by maintaining its state information and solution pool. Finally, we limit the CPLEX solver to a single thread per subproblem, because all subproblems are solved in parallel.

3) *Generator-specific values for ρ*

It is well-known that the parameter ρ in PH Step 4, Step 6, and Step 8 is important for obtaining acceptable performance for PH on stochastic mixed-integer programs. If ρ is too high, then all the scenarios agree after the first iteration, and the solution is usually severely sub-optimal. If ρ is too low, then PH may need too many iterations to come to an acceptable consensus. SLAC uses a modified idea from literature [8]. For each generator g that is off but eligible to turn on, we set $\rho_g = \max\{0.5, 0.5\,\overline{c_g} \cdot UT_g\}$, where $\overline{c_g}$ is the average cost of operating generator g, and UT_g is the minimum up-time requirement of generator g. Similarly, for each generator g that is on but eligible to turn off, we set $\rho_g = \max\{0.5, 0.5\,\overline{c_g} \cdot DT_g\}$, where DT_g is the minimum down-time requirement for generator g. These values are selected because they make ρ_g approximately proportional to the cost of committing or decommitting this unit at this time period. While this value of ρ_g for each committable/de-committable generator g results in good performance when economic trade-offs are in-play, it often does poorly when reliability trade-offs are being made. However, the larger ρ_g values needed for reliability trade-offs lead to bad performance when economic trade-offs are considered. The solution recovery heuristic, described next, makes up for this shortcoming when reliability trade-offs are considered.

4) *Solution recovery and iteration limit*

Due to time constraints, solution cycling, or other issues, practical implementation of PH needs an iteration limit ($\bar{k}$ in Step 10) and a recovery heuristic, which finds a here-and-now, or first stage, solution $\hat{x}$ that is feasible for every scenario s. The SLAC uses both: an iteration limit of $\bar{k} = 10$ for each SLAC problem, and a custom "slammer" heuristic, which we now describe. For problems with a pure-integer first stage, one can set ϵ in Step 10 to be an arbitrarily small positive number; we utilized 10^{-4} in our experiments.

Recall from Section II.A that each generator has at most a single first-stage commitment decision, associated with the time period in which LAC is first able to change its status. Let x be a vector of these first stage 0/1 commitment decisions associated with each generator, i.e., $x_g = 1$ means generator g is committed and $x_g = 0$ means generator g is *not* committed. Consider $\bar{x}^k$ from PH Step 7 at termination, which is a vector of average commitments over all subproblems, not necessarily 0/1 valued. We then build a first stage solution $\hat{x}$ as follows. If $\bar{x}_g^k > 0$ for generator g, then $\hat{x}_g := 1$ (i.e., we commit the generator), conversely, if $\bar{x}_g^k = 0$ for generator g, then $\hat{x}_g := 0$ (i.e., we do not commit the generator). On days with reliability issues (e.g., reserve shortfall), this simple heuristic significantly improved solution recovery over other approaches by committing any resource still needed in any scenario after 10 PH iterations.

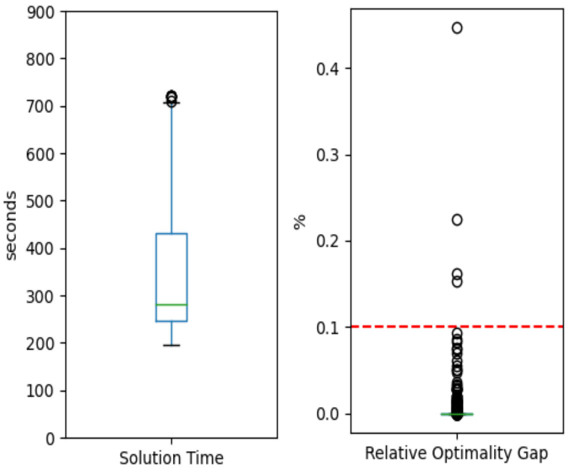

Fig. 2. Box-and-whiskers plots demonstrating SLAC solution time in seconds (left) and SLAC solution quality as measured by the computed relative optimality gap (right).

5) *SLAC Solver Solution Times*

Taken together, these PH implementation details and enhancements are shown to perform well against a total of 1,436 individual SLAC problems over 15 days, selected by MISO, representing different seasons and also days where the system was particularly stressed.

As mentioned above, we tracked the quantitative performance, both in terms of wall-clock time and solution quality, of each individual SLAC problem solved in the course of 15 days' worth of rolling horizon simulations for a total of 1,436 individual SLAC optimization problems, which are also used in Section IV to evaluate the qualitative performance of the SLAC. This gives a broad perspective of the computational performance of the SLAC solver, spanning different days, seasons, system configurations, and system stressors.

All computational evaluations were completed on a virtual machine provided by MISO, with 32 virtual CPUs and 256GB RAM. All SLAC solver runs used 20 concurrent threads and the CPLEX optimization solver version 20.1. Reported times are wall-clock times.

As can be seen in Fig. 2, all 1,436 SLAC instances are solved well-within the 15-minute (900 second) time limit established by MISO, and the majority are solved within five minutes (300 seconds). The solution time results in Fig. 2 include the time to read data from the disk, set up all twenty Pyomo models (one for each two-scenario subproblem), compute the objective value from the solution recovery heuristic, and write the full scenario solutions to the disk. It should be noted that these reported times are not overly optimized: better optimization of the Pyomo model build-time could halve the set-up time. Further, computing and writing the full SLAC solution (including recourse) is not strictly necessary for executing the here-and-now decision. Based on results from open-source stochastic unit commitment problems run at NREL, we would also expect further returns to parallelism with more compute resources [14].

6) SLAC Solver Solution Quality

We measure the solution quality using the typical “relative optimality gap” measure. That is, the relative optimality gap gap, for a given solution UB and given lower bound LB is:

$$gap = \frac{UB - LB}{UB}.$$

The team set a target solution quality, or relative optimality gap, of 0.1% for the SLAC, consistent with general practice. In Fig. 2, we detail in a box-and-whiskers plot the SLAC solution quality obtained over the 1,436 instances examined. As seen, all but four out of 1,436 SLAC problems (99.7%) are solved to a 0.1% optimality gap, and most (over 96%) meet a 0.01% optimality gap requirement; this is why the box-and-whiskers plot in Fig. 2 just has upper outliers. Finally, we note that this is a conservative estimate of the bound, as we obtained the lower bound LB at no additional computational cost as an outcome of PH Step 2. With more computational resources, a potentially better lower bound could be computed from each PH iteration [26], or other approaches for generating strong lower bounds could be applied [27]. Regardless, in every case the computed relative optimality gap was less than 0.5%, which is generally be considered an acceptable, if not good, solution.

IV. Rolling Horizon Simulations

A. Overview of Rolling Horizon

The interaction between the LAC and the single period real-time SCED is illustrated in Fig. 3. This figure illustrates three consecutive iterations of the LAC, which is solved every fifteen minutes and looks ahead with a 3-hour horizon. The SCED is solved every five minutes and looks ahead 10 minutes, only optimizing over a single interval. Our simulations simplify current practices by considering a SCED that is solved every fifteen minutes. LAC determines the commitment values that are used and fixed in the SCED. The SCED then determines the generation dispatch, which dispatches all generators and is used as the initial generation for each generator in the next iteration of the LAC. This process continues iteratively.

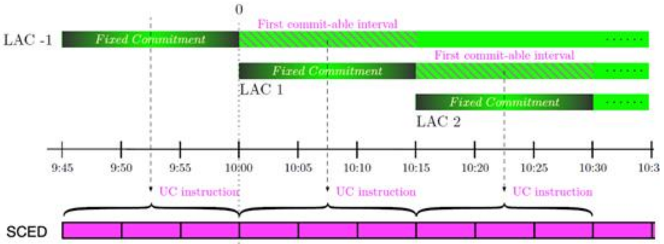

Fig. 3. Interaction between LAC/SLAC and SCED. Three consecutive LAC/SLAC iterations are shown. LAC/SLAC decide commitment statuses used in the SCED and the SCED determines the initial conditions used in the next iteration of the LAC/SLAC.

In this section we use the rolling horizon simulations to compare the performance of three different models of the LAC. The first is the deterministic LAC using the MISO forecast used in practice (termed the MISO LAC). The second is the deterministic LAC using the point forecast developed as part of this project led by ASU [15], which provides a forecast using a stochastic model that considers past MISO forecast performance as well as the relationship between past measured values and future power outputs (termed the ASULAC) [15]. The third is the Stochastic LAC with 40 scenarios (termed the SLAC). We use a few simplifications in the rolling horizon simulations in this section. First, the SCED intervals are assumed to be solved every fifteen minutes as opposed to the five-minute frequency used in practice. Second, we assume that the system operator follows the commitment decisions provided by the LAC/SLAC. Indeed, in practice the LAC and SLAC are only advisory tools and as a result the system operator may choose not to implement their suggested commitment decisions. Finally, we neglect the forecast update between the LAC/SLAC and the SCED, assuming that the same forecast information is available when solving the LAC/SLAC and the SCED.

The main limitation of our results follows from the fact that we are using historical data from 2018 and 2019. First, it is difficult to disentangle manual operator actions, such as emergency deployment and operator over-rides, from the historical data. To the extent that manual operator actions resolve unexpected events, we underestimate the value of SLAC, because the SLAC is intended to resolve these unexpected events itself. Second, these results are limited to the uncertainties that existed in the MISO system in the years 2018 and 2019. Indeed, we would expect the SLAC to become more valuable as the net-demand uncertainty increases. In particular, the SLAC would likely become more valuable in the future if the net-demand uncertainty increases from an increasing penetration of uncertain renewable generation. Third, the forecasts used in this study are only intended to capture uncertainty in net-demand and NSI. As a result, these forecasts are not intended to capture the many other uncertainties that an ISO/RTO experiences in practice. Future studies could extend this by additionally forecasting uncertainties not captured in our work.

In these results we only see improvements by the SLAC during days where the system is particularly stressed, otherwise the SLAC performs similarly to the two deterministic LAC models. This is perhaps in contrast with what might be expected with a stochastic model – in our experiments SLAC does not significantly increase production cost unless a reliability benefit is also observed. When SLAC provides improvements over LAC we typically observe these improvements in one of two ways. The first and most common observed improvement increases reliability by reducing constraint violations. More specifically, there are some days where the SLAC reduces transmission constraint violations and reserve constraint violations as compared to the deterministic LAC models but increases production costs by a small amount. The second improvement that we observe is decreases in production costs as compared to the deterministic LAC models. More specifically, there are some days where the SLAC solution significantly decreases production costs as compared to the deterministic LAC solutions while maintaining similar constraint violation levels.

B. Rolling Horizon Simulation Results

We ran simulations for 15 days throughout 2018 and 2019. These days were chosen to represent stressed and conservative operation days from different seasons throughout the year with possible constraint violations and higher production costs. Fig.

4 provides a box and whiskers plot that illustrates the relative total production cost, transmission violations and reserve violations for each of the 15 days studied. The plots represent the SLAC value relative to the MISO LAC value and a positive value indicates that the SLAC value is larger. Due to MISO data confidentially, only the relative values (not absolute values) are shown.

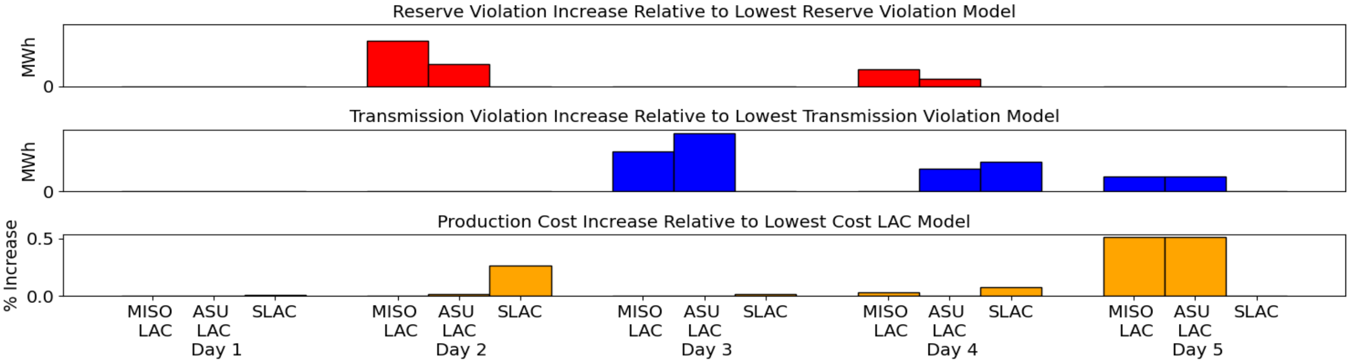

Fig. 5. Three bar plots comparing the performance of our three models for the first group of five days. For a given day the bottom bar plot shows the difference in production cost between each of the three models and the lowest cost model. The middle and top plots are similar. The middle plot represents the difference in transmission violation and the top plot represents the difference in reserve violation.

As is common, Fig. 4 illustrates that the SLAC production costs are typically slightly larger than the production costs resulting from the MISO LAC; however, this increase in production cost is typically small and is on the order of 0.01% of the total production costs in the system. Furthermore, outliers in the relative production cost tend to be symmetric around the mean. In other words, large savings in production cost are about equally as likely when using the SLAC or MISO LAC. On average, we conclude that the production costs do not significantly favor either the MISO LAC or the SLAC.

The transmission and reserve constraint violation plots in Fig. 4 illustrate that the benefits of SLAC are primarily realized by avoiding these constraint violations. None of the 15 days exhibited more reserve constraint violations when using the SLAC and only one of the 15 days exhibited more transmission constraint violations when using the SLAC, which will be further discussed in our analysis of Day 4 in Fig. 5. Furthermore, we occasionally see significant reductions in transmission and reserve violations when using the SLAC.

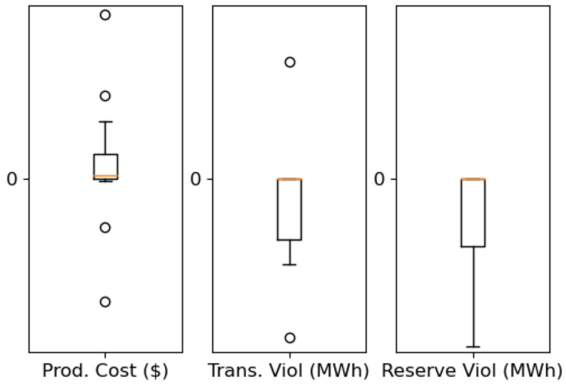

Fig. 4. Box and whiskers plot illustrating the relative total production costs, transmission constraint violations, and reserve constraint violations for each of the 15 days studied in this paper. Each plot represents the SLAC value relative to the MISO LAC value. A positive value indicates that the value is larger for the SLAC and a negative value indicates that the value is higher for the MISO LAC.

Fig. 5 illustrates the production cost and constraint violations for five characteristic days that were chosen to illustrate five distinctly different outcomes of using the SLAC as compared to deterministic LAC models. As compared to LAC, we observe days where SLAC performs similarly, reduces transmission violations, reduces reserve violations, reduces production costs, and shifts away from reserve violations towards transmission violations. Throughout the 15 days studied in the previous section, we never see a day in which the SLAC performs apparently worse than the LAC; however, we do see some days where the SLAC and LAC perform very similarly.

Fig. 5 illustrates the relative performance of each of the five days. For a given day, the bottom bar plot illustrates the difference in production costs between each of the three models. For each day, the model with the lowest production cost will have a plotted value of zero and the production costs for all other models are plotted with respect to that model. For example, the MISO LAC results in the lowest production cost for Day 2 and the SLAC increases these production costs by approximately 0.25% of the total production costs in the system. Similarly, the middle plot shows the increase in total transmission constraint violation relative to the model with the lowest transmission constraint violations. Finally, the top plot shows the increase in total reserve constraint violation relative to the model with the lowest reserve constraint violations. The violations are reported in units of MWh and represent the sum of violations throughout the day and specific numbers are not provide due to privacy concerns.

Day 1 represents a typical day that does not experience significant stress. In this situation the SLAC and both LAC models perform very similarly. Days like this often see a very small increase in production cost when using the SLAC in the amount of less than 0.02% of the total production costs in the system. This small increase in production costs is due to the SLAC prepositioning the system in a way that is more capable of accommodating stressful uncertainties if they were to arise; however, in Day 1 no such stressful uncertainties are realized.

During Day 2 the SLAC results in a significant reduction in reserve constraint violations at the expense of increasing production costs by approximately 0.25% of the total system production costs. Fig. 6 visualizes the preemptive action taken by the SLAC in Day 2 to avoid these reserve violations. On the left axis we see the net load and the 15-minute ahead available capacity plotted versus time using both the SLAC and the MISO LAC. The 15-minute ahead available capacity for a generator represents the minimum of its 15-minute ramping ability and its maximum output less its allocated reserve amounts. If the net load exceeds the 15-minute available capacity, then reserve violations will occur, intuitively representing the deployment of reserves. The available capacity is very similar using SLAC and MISO LAC and the two trajectories are often plotted on top of each other. It is apparent that there is sufficiently large available capacity early in the day; however, this abundance of available capacity diminishes during the upward net load ramp in the afternoon. This day represents a hot summer day and as a result, the system is very stressed during the peak. Indeed, this is when the reserve constraint violations occur. On the right axis we see two trajectories that represent the number of committed units as the simulations evolve in time. These trajectories match very closely for the MISO LAC and the SLAC; however, just before the reserve violations occur at the peak net load the SLAC commits additional units, which increases production costs but is able to avoid reserve constraint violations.

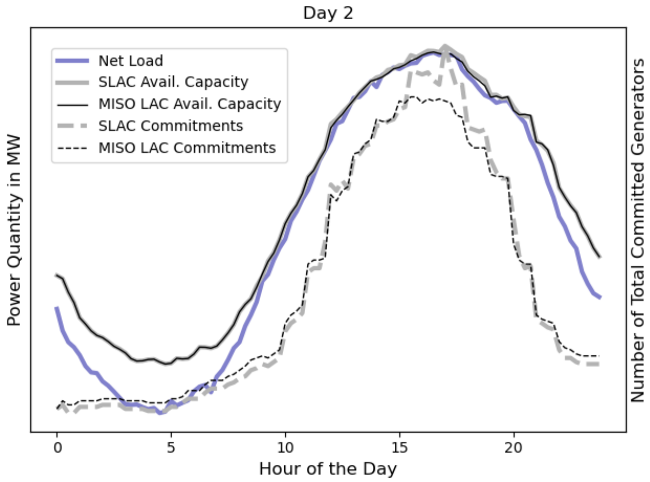

Fig. 6. SLAC vs MISO LAC available generation capacity compared to net load (solid lines, left axis) and number of committed units (dashed lines, right axis).

Day 2 is also a good example of how SLAC can avoid valuable reserve constraints pertaining to reserves that are not intended to accommodate net load fluctuation. Fig. 7 provides a breakdown of the reserve types that are violated during Day 2. This bar plot stacks the total ramp capability and spinning reserve violations in contrast to Fig. 5, which plots the relative violation across all reserve types. As is typical, most reserve violations pertain to the ramp capability requirement. This is because ramp capability reserve is partially intended to accommodate net load fluctuations and the penalty for violating ramp capability reserve requirements in the SCED is small relative to the other reserve requirements. If there is insufficient ramp capability reserve to accommodate the realized uncertainties, then reserve requirement violations will occur for spinning reserve, which exhibits higher violation penalties. Importantly, the SLAC completely avoids spinning reserve violations during Day 2, significantly improving the reliability of the power system as compared to the deterministic LACs.

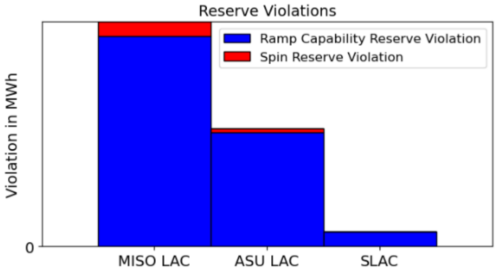

Fig. 7. Reserve violations for Day 2 displayed by reserve type.

Day 3 exhibits significant reduction in transmission violations when using the SLAC as compared to the deterministic LAC models with little to no increase in production costs and with no increase in reserve constraint violations. In this example, the SLAC's foresight was able to preposition the system in a way that clearly improves upon the deterministic LACs.

Day 4 represents a stressed day where all models result in significant penalties. Indeed, Day 4 represents the single outlier in Fig. 5 where the transmission violations of the SLAC are higher than the MISO LAC. This is because the SLAC trades-off transmission constraint violations for reserve constraint violations. In fact, LAC/SLAC objective function is designed to avoid reserve violations at the expense violating transmission constraints. The SLAC better achieves this objective because it can predict the reserve constraint violations earlier than the deterministic LACs. Furthermore, the improved deterministic forecast developed by ASU better avoids reserve constraint violations as compared to the MISO forecast.

During Day 5, the SLAC significantly reduces production costs by almost 0.5% of the total production costs as compared to the deterministic models and reduces the transmission constraint violations. In this example some uncertainty is realized that causes the deterministic LAC models to react quickly and at large cost. In contrast, the SLAC can predict these uncertainties in advance and preposition the system to accommodate these uncertainties at low cost. The ASU LAC and MISO LAC perform identically during Day 5.

V. Conclusions

This paper introduces the SLAC software prototyped and tested for the MISO look ahead commitment process. Study results on 15 MISO production days show that SLAC may bring economic and reliability benefits for stressed days. The computational performance with the customized progressive hedging method demonstrates the feasibility of real-world application. Rolling horizon simulations illustrate the performance benefit of using SLAC versus alternative deterministic LAC formulations. In particular, SLAC is shown to improve system reliability by reducing transmission and reserve constraint violations with little impact on production costs.

The uncertainty considered in this work specifically focuses on net-demand uncertainties and also intends to represent this uncertainty as it existed in the 2018 data. With increasing levels of renewable energy, the net-demand will likely become more uncertain and intermittent in the future. As a result, we would expect the value of the SLAC to increase in the future as compared to the deterministic LAC, and our results should represent an underestimate of the SLAC tool's future performance.

As an advisory tool, SLAC output can be translated into valuable information to the operator such as suggested commitments, optimal scheduling and dispatch of resources, reserve requirements at both locational and zonal resolutions, ramping availability and requirements. =.

VI. Acknowledgements

This work was supported by the United States Department of Energy under the Grant DE-AR0000696. The authors thank Jessica Harrison, John Harmon, Congcong Wang and Jason Howard from MISO for their support and valuable discussions.

VII. References

[1] Y. Chen, V. Ganugula, J. Williams, J. Wan and Y. Xiao, "Resource transition model under MISO MIP based Look Ahead Commitment," in *2012 IEEE Power and Energy Society General Meeting*, 2012.

[2] Y. Chen, Q. Wang, X. Wang and Y. Guan, "Applying robust optimization to MISO Look-Ahead commitment," in *2014 IEEE PES General Meeting | Conference & Exposition*, 2014.

[3] Q. P. Zheng, J. Wang and A. L. Liu, "Stochastic Optimization for Unit Commitment—A Review," *IEEE Transactions on Power Systems,* vol. 30, no. 4, pp. 1913-1924, 2015.

[4] M. A. H. Håberg, "Fundamentals and recent developments in stochastic unit commitment," *International Journal of Electrical Power & Energy Systems,* vol. 109, pp. 38-48, 2019.

[5] K. Kim and V. M. Zavala, "Large-Scale Stochastic Mixed-Integer Programming Algorithms for Power Generation Scheduling," , pp. 493-512, 2016.

[6] M. Tahanan, W. v. Ackooij, A. Frangioni and F. Lacalandra, "Large-scale Unit Commitment under uncertainty," *A Quarterly Journal of Operations Research,* vol. 13, no. 2, pp. 115-171, 2015.

[7] B. Carlson, Y. Chen, M. Hong, R. Jones, K. Larson, X. Ma, P. Nieuwesteeg, H. Song, K. Sperry, M. Tackett, D. Taylor, J. Wan and E. Zak, "MISO Unlocks Billions in Savings Through the Application of Operations Research for Energy and Ancillary Services Markets," *Interfaces,* vol. 42, no. 1, pp. 58-73, 2012.

[8] R. T. Rockafellar and R. J. B. Wets, "Scenarios and policy aggregation in optimization under uncertainty," *Mathematics of Operations Research,* vol. 16, no. 1, pp. 119-147, 1991.

[9] S. Takriti, J. Birge and E. Long, "A stochastic model for the unit commitment problem," *IEEE Transactions on Power Systems,* vol. 11, no. 3, pp. 1497-1508, 1996.

[10] S. M. Ryan, R. J.-B. Wets, D. L. Woodruff, C. Silva-Monroy and J.-P. Watson, "Toward scalable, parallel progressive hedging for stochastic unit commitment," in *2013 IEEE Power & Energy Society General Meeting*, 2013.

[11] C. Ordoudis, P. Pinson, M. Zugno and J. M. Morales, "Stochastic unit commitment via Progressive Hedging — extensive analysis of solution methods," in *2015 IEEE Eindhoven PowerTech*, 2015.

[12] B. Rachunok, A. Staid, J.-P. Watson, D. L. Woodruff and D. Yang, "Stochastic Unit Commitment Performance Considering Monte Carlo Wind Power Scenarios," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, 2018.

[13] K. Cheung, D. Gade, C. Silva-Monroy, S. M. Ryan, J.-P. Watson, R. J.-B. Wets and D. L. Woodruff, "Toward scalable stochastic unit commitment. Part 2: Solver Configuration and Performance Assessment," *Energy Systems,* vol. 6, no. 3, pp. 417-438, 2015.

[14] B. Knueven, D. Mildebrath, C. Muir, J. D. Siirola, J.-P. Watson and D. L. Woodruff, "A Parallel Hub-and-Spoke System for Large-Scale Scenario-Based Optimization Under Uncertainty," Optimization Online, 2020. [Online]. Available: http://www.optimization-online.org/DB_FILE/2020/11/8088.pdf.

[15] T. Werho, J. Zhang, V. Vittal, Y. Chen, A. Thatte and L. Zhao, "Scenario Generation of Wind Farm Power for Real-Time System Operation," arXiv, 16 June 2021. [Online]. Available: https://arxiv.org/abs/2106.09105.

[16] Midcontinent Independent System Operator, Inc., "BPM 002 - Energy and Operating Reserve Markets," [Online]. Available: https://www.misoenergy.org/legal/business-practice-manuals/.

[17] Y. Chen, P. Gribik and J. Gardner, "Incorporating post zonal reserve deployment transmission constraints into energy and ancillary service co-optimization," *IEEE Trans. on Power Syst.,* vol. 29, no. 2, pp. 537-549, March 2014.

[18] B. Wang and B. F. Hobbs, "Real-Time Markets for Flexiramp: A Stochastic Unit Commitment-Based Analysis," *IEEE Transactions on Power Systems,* vol. 31, no. 2, pp. 846-860, 2016.

[19] M. L. Bynum, G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson and D. L. Woodruff, Pyomo--optimization modeling in python, Third ed., vol. 67, Springer Science & Business Media, 2021.

[20] W. E. Hard, J.-P. Watson and D. L. Woodruff, "Pyomo: modeling and solving mathematical programs in Python," *Mathematical Programming Computation,* vol. 3, no. 3, pp. 219-260, 2011.

[21] Pyomo Developer Team, "Pyomo: Python Optimization Modeling Objects," 2021. [Online]. Available: https://github.com/Pyomo/pyomo.

[22] M. Bynum, A. Castillo, C. Laird, B. Knueven and J.-P. Watson, "EGRET: Electrical Grid Research and Engineering Tools," 2021. [Online]. Available: https://github.com/grid-parity-exchange/Egret.

[23] B. Knueven, J. Ostrowski and J.-P. Watson, "On Mixed-Integer Programming Formulations for the Unit Commitment Problem," *Informs Journal on Computing,* vol. 32, pp. 857-876, 2020.

[24] D. Woodruff, "mpi-sppy," 2021. [Online]. Available: https://github.com/Pyomo/mpi-sppy.

[25] L. D. Dalcin, R. R. Paz, P. A. Kler and A. Cosimo, "Parallel distributed computing using Python," *Advances in Water Resources,* vol. 34, no. 9, pp. 1124-1139, 2011.

[26] D. Gade, G. Hackebeil, S. M. Ryan, J.-P. Watson, R. J.-B. Wets and D. L. Woodruff, "Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs," *Mathematical Programming,* vol. 157, no. 1, pp. 47-67, 2016.

[27] N. Boland, J. Christiansen, B. C. Dandurand, A. C. Eberhard, J. T. Linderoth, J. R. Luedtke and F. Oliveira, "Combining Progressive Hedging with a Frank--Wolfe Method to Compute Lagrangian Dual Bounds in Stochastic Mixed-Integer Programming," *Siam Journal on Optimization,* vol. 28, no. 2, pp. 1312-1336, 2018.