Using Synthetic Traces for Robust Energy System Sizing

Sun Sun, Fiodar Kazhamiaka, Srinivasan Keshav, Catherine Rosenberg
University of Waterloo, Canada
sun.sun,fkazhami,keshav,cath@uwaterloo.ca

ABSTRACT

Due to the inherent randomness of both solar power generation and residential electrical load, jointly sizing solar panel and storage capacity to meet a given quality-of-service (QoS) constraint is challenging. The challenge is greater when there is limited representative historical data. We therefore propose generating *synthetic* solar and load traces, corresponding to different realizations of the underlying stochastic processes. Specifically, we compare the effectiveness of three generative models: autoregressive moving-average (ARMA) models, Gaussian mixture models (GMMs), and generative adversarial networks (GANs) – as well as two direct sampling methods – for synthetic trace generation. These traces are then used for robust joint sizing by a technique described in recent work. Extensive experiments based on real data show that our approach finds robust sizing with only one year's worth of hourly trace data. Moreover, assuming that solar data are available, given a database of load traces, we demonstrate how to perform robust sizing with access to only twelve data points of load, one for each month of one year.

CCS CONCEPTS

• **Mathematics of computing** → **Probability and statistics**.

KEYWORDS

Robust sizing, Solar, Storage, Generative models

1 INTRODUCTION

With the rapid decline in the price of solar photovoltaic (PV) systems, commercial and residential buildings are increasingly using roof-mounted solar panels to generate energy. However, solar and load are both intrinsically uncertain, so solar generation never exactly matches consumption. This mismatch is balanced either using storage devices (e.g., batteries) [18] or by purchases from and sales to the electrical grid. Both storage devices and grid purchases add to system cost, so it is important to jointly size solar generation and storage systems to minimize this cost.

In this work, for simplicity, we consider the case of sizing solar and storage resources (or *sizing pairs*) for an off-grid residential or commercial building. Here, if solar panels and storage are undersized the load cannot be fully met all the time, which inevitably decreases the comfort level of building occupants. On the other hand, if they are over-sized the capital cost is higher, which is also undesirable. The objective of our work, therefore, is to minimize the capital cost of installing solar panels and storage subject to system operational constraints and a quality-of-service (QoS) constraint on the unmet load.

Determining an optimal sizing pair for a given building is challenging because we typically have access to only limited data from the past and we want to size for the *future*. Even assuming that the past data well represents the future, the QoS actually achieved depends on the values realized by the solar and load processes. Thus, we can only meet probabilistic QoS constraints. For concreteness, a typical probabilistic QoS constraint is "the percentage of the unmet load during a fixed period (e.g., any consecutive period of 100 days) should not exceed a certain threshold (e.g., 5%)". We call such a fixed period the *QoS period*.

Prior work by Kazhamiaka *et al.* [27] presented robust and practical techniques to compute a sizing pair. Their approach was found to work well when three years of hourly solar and load traces were available. In contrast, we consider situations where there is access to (a) only a single year of hourly trace data, and (b) when the data is in the form of daily or monthly aggregates of solar and load. As in prior work, to account for uncertainty, we consider a *scenario-based* robust design approach. By a scenario, we mean a possible realization of future events (That is, it is a trajectory of the underlying stochastic process.). To deal with the problem of having a limited horizon of available data, we empirically compare several distinct approaches to generating synthetic data traces. Moreover, to accommodate consumers with access to only aggregated data–the common but more challenging case–we propose using a database consisting of complete hourly data from different consumers, using traces from the most-similar user in the database to parametrize our generative models.

Our key contributions are:

- To combat the uncertainty of solar and load as well as the limited size of historical data, we propose to first generate synthetic traces using generative models and then feed them to an existing robust-sizing framework. We empirically compare the performance of three distinct generative models (ARMA, GMMs, and GANs) with two direct sampling methods based on real dataset in terms of the capital cost incurred by the sizing system and the number of unmet QoS constraints in testing years.
- We find that compared to the other two generative models, in our dataset an ARMA-based approach can always meet QoS constraints in testing years with only one year's worth of historical trace data. Compared to the direct sampling methods, the ARMA-based approach generally incurs a lower capital cost.
- Assuming solar data and a database of typical consumer loads are available, our approach can determine a robust sizing with

e-Energy '19, June 25–28, 2019, Phoenix, AZ, USA

ACM ISBN 978-1-4503-6671-7/19/06…$15.00
https://doi.org/10.1145/3307772.3328306

access to only twelve monthly load data points, which is the typical case. In particular, the corresponding costs with the monthly load data are 2% to 14% higher than those computed with hourly data, which can be interpreted as the cost of having incomplete information.

The organization of rest of the paper is as follows. In Section 2 we briefly summarize related work. In Section 3 we describe the system model and our research focus. In Section 4 we sketch prior work on a robust sizing framework, and in Section 5 we extend it to accommodate aggregated data. Our generative models are presented in Section 6, and the experimental results are shown in Section 7. We discuss our work and conclude in Sections 8 and 9, respectively.

2 RELATED WORK

2.1 Sizing Approaches

Mathematical approaches to solar and storage sizing fall into three general classes: mathematical programming, simulation, and analytical methods. In the first class, sizing is formulated as an optimization problem. Here, both the sizes of solar panels and storage, as well as the system operation–such as choosing when to charge or discharge storage, and by how much, are treated as optimization variables (e.g., [15, 16, 20, 25, 30]). For example, Chen *et al.* [16] study storage sizing in the context of the daily operation of a microgrid. They formulate a mixed-integer linear program, where the renewable energy is an input forecast using historical data. Similarly, Jabr *et al.* [25] formulate a two-stage robust optimization for storage sizing on transmission networks and Cervantes *et al.* [15] compute the joint sizing of solar panels and storage using a mixed-integer stochastic optimization problem with 24-hour planning horizon. Although we also formulate the problem as an optimization problem, we find that using simulations to compute sizing pairs is faster in practice.

Simulation has also been used in prior work to size solar and storage systems. Simulations require the pre-specification of the system operation policy. Compared to optimization approaches, these operation policies are generally non-optimal. However, simulations can use more-accurate, complex, non-linear storage models and the control rules specified are easy to implement in practice. Studies using this approach include References [12, 27, 28]. We also use simulations in our work. In contrast to prior work, the simulations use synthetic load and solar traces.

Analytical approaches to storage sizing treat storage as an energy buffer analogous to a data buffer in a communication network. Recently, several mathematical techniques that were originally used in data networking have been applied to energy storage, for example, Lyapunov optimization [34] and stochastic network calculus [26]. In particular, Lyapunov optimization technique has been exploited by Sun *et al.* and Guo *et al.*[23, 41] for storage charging/discharging control under uncertainty. The work that is closest to ours, by Kazhamiaka *et al.* [27] uses stochastic network calculus for robust sizing of solar panels and storage. This approach is also compared with optimization and simulation methods. Prior work using mathematical approaches either focus on storage sizing alone or assume the availability of rich historical data (i.e., several years of hourly data). We instead focus on joint robust sizing of solar panels and storage with limited access to historical trace data.

2.2 Generative Models

In renewable-energy energy systems, many operational decisions involve uncertainty due to the intrinsic randomness of renewable generation and load. Typically, we have access to traces of sample paths generated by the underlying random processes and sizing has to be robust in that the QoS constraint should be met for a large percentage of possible sample paths or realizations. Scenario-based methods generate many such realizations and ensure that the sizing is suitable for all or a large percentage of them. This approach is commonly used in energy applications such as unit commitment and frequency regulation [24, 36, 44**?**]. However, the performance of scenario-based decisions depends on the number of scenarios and how well they follow the distribution of real data.

Commonly-used approaches for scenario generation include time-series approaches, probabilistic approaches, and neural network (NN)-based approaches. These generative models are usually built based on historical data and then used to produce synthetic samples or scenarios. In our work, we develop generative models using all three methods. We discuss prior work using these methods next.

In a time-series approach, models such as autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA), capture the time correlation between data points [19, 33]. In contrast, with a probabilistic approach, data sequences are characterized by probabilistic models such as Gaussian mixture models [35, 39, 42], the Weibull distribution [14, 31, 38], and Markov chain [27].

Unlike the previous two approaches, which require either statistical or probabilistic assumptions that may not hold in practice, a NN-based approach is data-driven. They have been found to outperform the other two approaches when given access to rich data input [10]. NN-approaches include wavelet neural networks, Bayesian neural networks, multi-layer perceptrons, and radial basis function networks. A systematic review on recent developments of NN applications in solar and wind energy systems can be found in References [10, 43].

More recently, generative adversarial networks (GANs) [9, 22, 32] have been widely adopted for generating images in computer vision and broadly in the machine learning community. They have been shown to outperform several state-of-the-art machine learning approaches in some contexts. For example, Chen *et al,* [17] use GANs to generate data for multiple renewable resources, and also to generate data conditioned on different events. Given the recent interest in this approach, we also use GANs to generate scenarios and compare the performance of this approach with those generated using classical (ARMA and GMM) approaches.

3 SYSTEM MODEL

In this section, we describe the system model and formulate an optimization problem for optimal and joint sizing of solar and storage under an ideal case where the realization of solar and load is available. We then focus on robust sizing with limited data.

3.1 Notation

The setting for our work is a commercial or residential building that is equipped with both solar panels and an energy storage device. Denote the capacity of the storage device by C (in kWh) and the

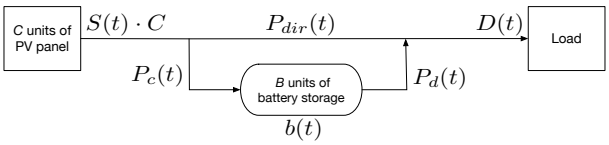

Figure 1: System operation at time slot t

maximum generation capacity of solar panels by B (in kW). Let the price of one storage cell and one solar panel be π_C (in \$/kWh) and π_B (in \$/kW), respectively. Then the capital cost of installing the solar panels and battery is $B\pi_B + C\pi_C$.

Consider a time-slotted system with time periods indexed by t. Denote the length of each time slot by T_u (e.g., one hour), the horizon (i.e., the total number of time slots under study) by T, the energy state of the battery at the end of the t-th time slot by $E(t)$, and its initial energy state by U. Denote the solar power generated at time slot t by $S(t)$; the load power at time slot t as $D(t)$, and the charging and discharging powers in time slot t by $P_c(t)$ and $P_d(t)$, respectively.

3.2 System Operation and Problem Formulation

We now discuss how to formulate the joint sizing problem as an optimization problem. Recall that to solve this problem we would need to accurately forecast hourly solar and load values for the lifetime of the equipment, approximately 25 years. Although this is impossible in practice, formulating the problem in this way allows us to be precise in specifying our optimization goal.

The system operation in each time slot is depicted in Fig. 1. In each slot, part of the solar generation, denoted by $P_{dir}(t)$, can be used directly to match (possibly partly) the load, while another part, denoted by $P_c(t)$, can be used to charge the battery. The storage operation is modeled using linear Model 1^* from [29], which has been shown to be more accurate than the commonly used linear model, which ignores the energy limits' dependency on the charging/discharging power. The distinguishing feature of Model1^* is that the effective lower (resp. upper) bound on $E(t)$ varies linearly with $P_d(t)$ (resp. $P_c(t)$) and B. Besides solar generation, the load can also be supported in part or totally by storage discharging with a power denoted by $P_d(t)$. It is possible that there is some unmet load, represented by $\delta_t(t)$, if the power from both solar and storage is insufficient.

We now state the optimization problem. Consider an *ideal* case in which the realization of $\{S(t), D(t)\}_{t=1}^{T}$ and the storage parameters are provided. The objective of the system is to minimize the capital cost subject to the battery operational constraints and a QoS constraint on the unmet load. For concreteness, suppose that we require that the percentage of the unmet load during a fixed period (e.g., 100 days) not exceed a pre-designed threshold denoted θ (e.g., 5%). Such a fixed period T is called the *QoS period*. In practice, the value of T is specified by consumers, and the higher the value of T the more tolerant the system. If consumers have more stringent requirement on the system, we can set a smaller value of T (e.g., 30 days or 10 days). Then, in the ideal case, the optimization problem can be formulated as follows:

$$\begin{aligned} \min \quad & B\pi_B + C\pi_C \\ s.t. \quad & P_c(t) + P_{dir}(t) \le S(t)C, \forall t \end{aligned} \tag{1}$$

$$P_{dir}(t) + P_d(t) = D(t) - \delta(t), \forall t \tag{2}$$

$$E(0) = U \tag{3}$$

$$E(t) = E(t-1) + P_c(t)\eta_c T_u - P_d(t)\eta_d T_u, \forall t \tag{4}$$

$$u_1 P_d(t) + v_1 B \le E(t) \le u_2 P_c(t) + v_2 B, \forall t \tag{5}$$

$$0 \le P_c(t) \le B\alpha_c, \forall t \tag{6}$$

$$0 \le P_d(t) \le B\alpha_d, \forall t \tag{7}$$

$$P_c(t)P_d(t) = 0, \forall t \tag{8}$$

$$\sum_{t=1}^{T} \delta(t) \le \theta \sum_{t=1}^{T} D(t) \tag{9}$$

$$B, C, P_{dir}(t), \delta(t), E(t) \ge 0, \forall t, \tag{10}$$

where the optimization variables are $B, C, P_c(t), P_d(t), P_{dir}(t), \delta(t), E(t), \forall t$. In constraint (4), $\eta_c \in (0,1)$ and $\eta_d \in (1,\infty)$ are storage charging and discharging efficiency parameters, respectively. In constraint (5), u_1, v_1, u_2, v_2 are coefficients in the linear storage model. In constraints (6) and (7), α_c and α_d are the charging and discharging rate limit, respectively.

3.3 Our Focus: Robust Sizing with Limited Data

The problem above is a non-linear optimization problem. To solve it, storage parameters can be derived either from the storage specification sheet published by the manufacturer or a experimentally-derived measurement trace. The main challenge lies in the uncertainty of solar and load. In other words, when we need to determine the sizing pair B and C for a given system, we are unaware of the actual values that will be taken by solar and load. In practice, we usually have limited historical data, especially for load, for at most one or two years. Predicting the values of solar and load is impractical with such limited data.

For example, consider the QoS period as 100 days and the length of each time slot $T_u = 1$ hour. Then, to carry out this optimization, we need to forecast $2 \times 24 \times 100 = 4800$ values for solar and load, a challenging and error-prone task. Note that since the optimal sizing pair depends on the realization (i.e., actual values assumed by) of $S(t)$ and $D(t)$, the low quality of $S(t)$ and $D(t)$ estimates would inevitably lead to a poor sizing decision.

What we propose, instead, is a robust sizing design based on *scenario generation*. The main idea is to compensate for a lack of input traces by generating many synthetic solar and load scenarios that exhibit a variety of patterns, but nevertheless share the same statistics as historical data. Once synthetic scenarios are produced, we can use them to find scenario-based sizing pairs. However, it is important to realize that because the underlying model of future data may not be exactly the same as that of historical data, it is critical to adopt a robust sizing approach [27] to account for model uncertainty.

We focus on a practical sizing case in which consumers only have limited data for the following two setups:

(1) Assume that consumers have one-year of hourly data for solar and load.
(2) Assume a more common, but even more challenging case, in which consumers only have access to aggregate data (e.g., monthly/daily data).

In both cases, consumers wish to size their solar and storage system such that the load constraint (9) can be met for any future QoS

period. For the first case, we propose three different generative models and two direct sampling methods for generating synthetic data, and then feed data into an existing robust sizing framework. For the second case, we use a database of available traces to find the closest-matching consumer for the target consumer.

Note that generative models are used to produce synthetic data, which are then used for sizing. Thus, below we first describe the robust sizing method in Section 4. We extend this method to accommodate aggregate data in Section 5. Finally, generative models for synthetic trace generation are discussed in Section 6.

4 ROBUST SIZING WITH CHEBYSHEV BOUNDS

As we noted in the last section, robust sizing is necessary because solar and load observed in the future is unlikely to be identical to historical observations. Additionally, due to the limited size of historical data, the extracted historical model may be inaccurate. The idea of robust sizing is to explicitly account for model uncertainty and inaccuracy.

4.1 Sizing Curves

Instead of solving an optimization problem, we assume a *storage operating policy* by which solar generation is used to maximally support load. This operating policy may be non-optimal in the sense of the the optimization problem in Section 3.2, but is rational and easy to implement. Specifically, in each time slot, solar generation is first allocated to meet load. If this is insufficient, we then discharge storage to support load under its operational constraints (4)(7)(8). However, if there is sufficient solar generation, the excess is used to charge storage under the storage operational constraints (4)(6)(8).

We make the realistic assumption that that the storage capacity B can only take one of b different values from the set $\{B_1, B_2, \cdots, B_b\}$ with the elements in ascending order; similarly, the size of solar panels C can only take on one of c different values from the set $\{C_1, C_2, \cdots, C_c\}$ with the elements in ascending order. Consequently, there are at most bc different combinations of the *sizing pairs* (B, C). We denote a pair of solar and load traces $\{S(t), D(t)\}_{t=1}^{T}$ as a *scenario*. Given a scenario and an operating policy, we define a *sizing curve* to be the set of sizing pairs (B_s, C_s) on the Pareto frontier; that is, given a particular value of B (resp. C), no smaller value of C (resp. B) meets the QoS criterion.

Note that solving the optimization problem in Section 3.2 allows us to compute an *optimal sizing pair* (B, C) for any scenario as the least-cost point on the corresponding sizing curve. However, using this approach to compute a sizing pair in practice would require us to accurately predict the future. Instead, we adopt the recommendation in Reference [27], which is to (a) generate a set of potential future scenarios; (b) use simulations to find a sizing curve for each scenario; (c) denote the least-cost point on it as the sizing pair for that scenario, then (d) use a multivariate Chebyshev bound to find a *robust optimal sizing pair*.

Note that Kazhamiaka *et al.* [27] study only Gaussian mixture models (GMMs) for generating scenarios. In contrast, we discuss several methods for scenario generation in Section 6. Moreover, they choose the robust optimal sizing pair as the one with the least cost from the upper right quadrant of the ellipsoid representing the multivariate Chebyshev bound. However, in our experimental evaluation, we found that this approach often leads to over-sized values of B and C, and sometimes the least cost point in the upper right quadrant is clearly under-sized compared to other points on the ellipse (i.e., has a lower value of both B and C compared to other points on the ellipse, and hence dominated). Therefore, we replace the multivariate bound with a univariate Chebyshev bound on B and C [1][1].

4.2 Univariate Chebyshev Bounds

We begin by describing how to build a univariate Chebyshev bound on C (the univariate Chebyshev bound on B is similar). Given N potential future scenarios, we use simulations to construct N sizing curves denoted by $K_n, n \in \{1, 2, \cdots, N\}$. For curve K_n and a value of B as B_i, we denote the corresponding Pareto-optimal value of C by $K_n(B_i)$. Conversely, if the value of C is C_j, we denote the corresponding Pareto-optimal value of B under K_n by $K_n^{-1}(C_j)$.

For a value B_i, denote the set of all Pareto-optimal values of C by L_{B_i}, i.e.,

$$L_{B_i} \triangleq \{C : K_n^{-1}(C) = B_i, n = 1, \cdots, N\}. \tag{11}$$

Let $N_{B_i} = |L_{B_i}|$. Note that for a certain value B_i, for some sizing curves, there may be no feasible corresponding C_i. Thus $N_{B_i} \leq N$.

Using the points in L_{B_i}, we can compute a sample Chebyshev bound on C based on Theorem 1 in [40] as follows:

$$p\left(|C - \mu_{C,B_i}| \geq \lambda \sigma_{C,B_i}\right) \leq \min\left\{1, \frac{1}{N_{B_i}+1}\left\lfloor \frac{(N_{B_i}+1)(N_{B_i}^2 - 1 + N_{B_i}\lambda^2)}{N_{B_i}^2 \lambda^2} \right\rfloor\right\}, \tag{12}$$

where $\lfloor \cdot \rfloor$ is the floor operator, μ_{C,B_i} is the sample mean, σ_{C,B_i} is the unbiased estimate of standard deviation, and $\lambda > 0$ is a QoS parameter. Under the assumption that the values of C in L_{B_i} are i.i.d. samples under some distribution, the above inequality provides a probability on how the random variable C can deviate from its sample mean.

For robust design, the value of C is set to be

$$C_{B_i}^* = \mu_{C,B_i} + \lambda \sigma_{C,B_i}, \tag{13}$$

where the value of λ is determined as the smallest one satisfying

$$\frac{1}{N_{B_i}+1}\left\lfloor \frac{(N_{B_i}+1)(N_{B_i}^2 1 + N_{B_i}\lambda^2)}{N_{B_i}^2 \lambda^2} \right\rfloor \leq 1 - \gamma. \tag{14}$$

In this approach, a parameter γ denotes a confidence measure, which controls the level of robustness. In particular, the higher the value of γ, the more robust the sizing design.

Following this procedure, we find $C_{B_i}^*$ for each B_i. The pairs of points $(B_i, C_{B_i}^*)$ are interpolated to form a curve, which we call the Chebyshev curve on C. Similarly, we construct the Chebyshev curve on B by considering each fixed value of C, and then create the Chebyshev bound on B similar to the inequality (12). The upper envelope of the Chebyshev curves on B and C represents the system sizings that are robust with respect to both B and C with the confidence measure γ. The final robust sizing pair is determined as the one with the minimum capital cost on the upper envelope.

[1]Although the full text of this submission is accessible to reviewers, it has been anonymized for double-blind reviewing.

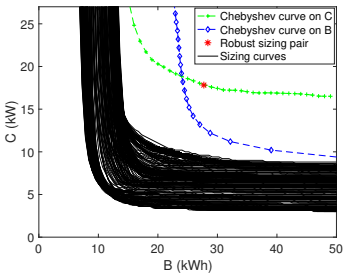

Figure 2: Generation of a robust sizing pair for $T = 100$ days: 365 sizing curves (boosting)

Consider the QoS period T equal to 100 days. As an example, in Fig. 2, based on 365 sizing curves, we construct the Chebyshev curves on B and C and then find a robust sizing pair on the upper envelope. The scenarios in Fig. 2 are generated by boosting, described in Section 6.4. The parameter γ is set to be 0.97.

5 SIZING FOR CONSUMERS WITH AGGREGATED DATA

Thus far, we have assumed that consumers have access to at least one year of hourly solar and load data (i.e, 24×365 data points for solar and load respectively). In practice, hourly data is rarely available. Moreover, some consumers are only willing to provide aggregated data (e.g., monthly/daily) due to privacy concerns. Such aggregated data can be easily obtained from monthly electricity bills or from utility providers' consumer-facing websites.

Note that the main challenge is to generate *synthetic* hourly data that is representative of future load patterns of the target consumer; once this is done, the trace pair can be fed into our robust sizing framework described in Section 4. To address this challenge, we match aggregated load values with loads in a database of hourly load traces, summarized here:

Step 1: *Find the most similar consumer in the database*. Assume that we have a database $\mathcal{S}$ consisting of N consumers with hourly solar and load data. Using the database, we find the traces most similar to the target consumer in terms of aggregated load consumption and solar generation respectively. To eliminate issues with scaling, aggregated data are first normalized to lie in [0,1].

Specifically, let the monthly aggregated data be provided by a target consumer U. Denote the normalized load consumption and solar generation of consumer U in the m-th month as $l_{u,m}$, and $s_{u,m}$, respectively. Analogously, denote the normalized load consumption and solar generation of consumer $i \in \mathcal{S}$ as $l_{i,m}$, and $s_{i,m}$, respectively. Then, the most similar consumer is defined as:

$$V_l = \arg\min_{i \in \mathcal{S}} \sum_{m \in \mathcal{M}} |l_{i,m} - l_{u,m}| \quad \text{(load)} \tag{15}$$

$$V_s = \arg\min_{i \in \mathcal{S}} \sum_{m \in \mathcal{M}} |s_{i,m} - s_{u,m}| \quad \text{(solar)}, \tag{16}$$

where $\mathcal{M}$ is the set of all months.

Step 2: *Generate synthetic hourly data*. After obtaining the most similar consumers V_l and V_s, we generate synthetic hourly data based on a generative model (see Section 6) of V_l and V_s.

Step 3: *Adjust the magnitude of synthetic hourly data for the target consumer*. Specifically, the generated hourly data in the m-th month in Step 2 is multiplied by $l'_{u,m}/l'_{V_l,m}$, where the superscript $'$ indicates the original un-normalized data. That is, the hourly scaling factors are approximated by the monthly ones. As another example, for daily aggregated load data, the generated hourly data on the d-th day of the m-th month in Step 2 are multiplied by $l'_{u,dm}/l'_{V_l,dm}$, where $l'_{u,dm}$ is the un-normalized load on the d-th day of the m-th month.

6 SYNTHETIC TRACE GENERATION

In this section we introduce three generative models for generating synthetic solar and load traces: autoregressive moving-average (ARMA) models, Gaussian mixture models (GMMs), and generative adversarial networks (GANs). For comparison, we also consider two simple direct sampling (boosting) methods. We choose to model solar and load separately (as opposed to jointly) to reduce model complexity. The three generative models are very different in nature. For ARMA, we parameterize one model per month, i.e., twelve models for one year. For GMMs, we parameterize one model per clock time (e.g., 9am or 11pm) per season, i.e., 96 models for solar and double that for load, which we explain in Section 6.2. For GANs, we parameterize a unique model for one year.

6.1 Autoregressive Moving-Average Models

The time series of solar generation and load are generally non-stationary due to diurnality, seasonality, and long-term trends [27]. For a non-stationary time series $\{X(t)\}$, a classical decomposition model of $X(t)$ can be represented by:

$$X(t) = m(t) + s(t) + Y(t), \tag{17}$$

where $m(t)$ is a slowly changing trend component, $s(t)$ is a seasonal component with a period d that satisfies $\sum_{j=1}^{d} s(j) = 0$, and $Y(t)$ is a stationary random noise with mean zero [13]. What we suggest is to characterize $\{Y(t)\}$ using ARMA models, which have been widely used for modeling stationary time series data in energy applications. In particular, an ARMA(p, q) model describes the following type of time series:

$$Y(t) = \sum_{i=1}^{p} \phi_i Y(t-i) + \epsilon(t) + \sum_{i=1}^{q} \theta_i \epsilon(t-i),$$

where ϵ_t is white noise.

To extract ARMA models from real data, we first divide data into a small number of periods (e.g., seasons or months), and then we build an ARMA model for each period. To get the stationary times series $\{Y(t)\}$, we need to first remove the deterministic components $m(t)$ and $s(t)$.

First consider the removal of the seasonal component $s(t)$. The periods of a time series can be easily observed in the frequency domain by plotting the periodogram [5]:

$$I(\omega_k) = \frac{1}{T}\left|\sum_{t=1}^{T} x(t)\exp\{-2\pi(t-1)\omega_k i\}\right|^2,$$

where $\{x(t)\}$ are the observations of the time series, $\omega_k = k/T, k = 1, \cdots, [T/2]$ with $[\cdot]$ denoting the largest integer less than or equal to the number inside, and i is the imaginary unit.

As an example, in Fig. 3, we show the periodogram of one-year hourly data of solar and load downloaded from the Pecan Street

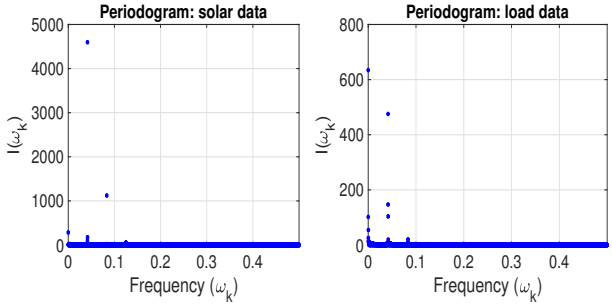

Figure 3: Periodogram of one-year hourly solar and load data

Dataport (in Austin, TX) [4]. The dominant period can be calculated by $1/\omega_k$, where ω_k corresponds to the frequency leading to the dominant peak in the spectral density $I(\omega_k)$. Based on this analysis, for solar data, we find the dominant period to be 24 hours (the daily cycle). For load data, the dominant peak is derived as one year. Nevertheless, since we focus on having access to only one-year of historical data, we revert to the secondary-peak period of 24 hours.

To remove the seasonal component, we use a moving average technique. Specifically, let $d = 2q$, i.e., $q = 12$. We first estimate the trend component as follows:

$$\hat{m}(t) = (0.5x(t-q) + x(t-q+1) + \cdots + 0.5x(t+q))/d,$$

where $q < t \leq T - q$. Next, we estimate the seasonal component $s(t)$. For each $k = 1, \cdots, d$, consider the de-trended data points $\{x(k+jd) - \hat{m}(k+jd), q < k+jd \leq T-q, j = 1, \cdots, d\}$, and denote their average by w_k. Then the seasonal component s_k can be estimated as follows:

$$\hat{s}_k = \begin{cases} w_k - \frac{1}{d}\sum_{i=1}^{d} w_i, & k = 1, \cdots, d, \\ \hat{s}_{k-d}, & k > d, \end{cases} \tag{18}$$

which can be verified to satisfy the constraints in equation (17). The de-seasonalized data $\{d(t)\}$ can be obtained as

$$d(t) = x(t) - \hat{s}(t), \quad t = 1, \cdots, T.$$

We then remove the trend component from the de-seasonalized data $\{d_t\}$ by exponential smoothing. Specifically, we re-estimate the trend component of $\{d_t\}$ as follows:

$$\hat{m}(t) = \alpha x(t) + (1-\alpha)\hat{m}(t-1), \tag{19}$$

and $\hat{m}(1) = x(1)$. The parameter $\alpha \in (0, 1)$ needs to be tuned based on data. The estimate of the residual time series that we model is $\hat{y}(t) = x(t) - \hat{m}(t) - \hat{s}(t)$.

To determine the values of p and q in ARMA, a common practice is to try different combinations of (p, q) and choose the one with the minimum Bayesian information criterion (BIC) or Akaike information criterion (AIC). To generate synthetic time sequence, we first use the derived ARMA model to generate the noise time series $\{y(t)\}$, then add $\hat{m}_t$ and $\hat{s}_t$ back to y_t to get the final time sequence.

6.2 Gaussian Mixture Models

The distribution of a GMM can be represented by $\sum_i^I \pi_i \mathcal{N}(\mu_i, \sigma_i^2)$, where I is the number of Gaussian components, π_i is the probability associated with the i-th Gaussian component, μ_i is the mean and σ_i^2 is the variance of the i-th Gaussian component. Empirical distributions of solar and load data usually exhibit multiple peaks as

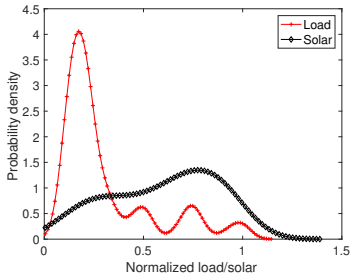

Figure 4: Empirical distribution of normalized load on a working-day and solar power in the Fall season, from 12 – 13pm

exhibited in Figure 4, consistent with a mixture model such as a GMM. In addition, GMMs have been extensively used for solar and load modeling in prior work [8, 11, 35, 39, 42].

For data modeling, we divide the annual historical data of solar and load into four seasons (i.e., spring, summer, fall, and winter). For each season we then build a GMM for each hour. Using this method, we implicitly assume independence between time slots. For load data, we additionally separate working days and non-working days since load consumption during working days is generally higher. Therefore, there will be $24 \times 4 = 96$ GMMs for solar, and $2 \times 24 \times 4 = 192$ GMMs for load. Such GMMs are also used in Reference [27] to generate solar and load data.

6.3 Generative Adversarial Networks

GANs have been widely applied in computer vision and machine learning areas for generating images [9, 22, 32, 37]. We now present a brief sketch of their working. The original GANs proposed in [22] are composed of two adversarial players: a generator (denoted by G) and a discriminator (denoted by D), playing a minmax game. The input of G are samples from a prior distribution denoted by p_z (e.g., Gaussian), and the output $G(z; \theta_g)$ are generated samples in the real data space, where θ_g denote the network parameters. Denote the underlying distribution of historical data by p_{data}. Unlike GMMs, GANs do not explicitly learn the expression of p_{data}. Denote the distribution of the generated samples as p_g. In contrast, the input of D can be either real samples or generated samples from G, and the output $D(x; \theta_d)$ is the probability of the sample coming from the real data distribution p_{data}, where θ_d denote the network parameters. The objective of G is to fool D by producing realistic samples, and the corresponding optimization is formulated as follows:

$$\min_G \mathbb{E}_z[\log(1 - D(G(z)))]. \tag{20}$$

In contrast, D aims to reject samples coming from G. The networks of G and D are usually deep neural networks, and are simultaneously trained with the overall objective function as follows:

$$\min_G \max_D V(D, G) = \mathbb{E}_x[\log D(x)] + \mathbb{E}_z[\log(1 - D(G(z)))].$$

For illustration, in Fig. 5, we show a schematic diagram of GANs for generating synthetic data. It can be proved that at the Nash equilibrium $p_g = p_{\text{data}}$. That is, G can produce samples exactly following the data distribution p_{data}, and thus D cannot distinguish between generated samples and real samples.

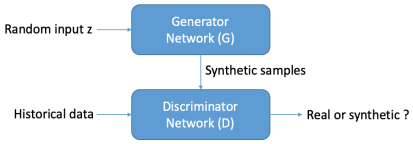

Figure 5: Architecture of GANs for generating synthetic data.

The above GANs were proposed by Goodfellow *et al.* [22]. It was shown that it can suffer from 'mode collapse,' which causes low output diversity. This is undesirable in our case, since we hope to simulate as many patterns of historical data as possible. The objective of G in the expression (20) can be understood as minimizing the Jensen-Shannon divergence between p_{data} and p_g. In [9], Wasserstein GANs (WGANs) is proposed in which the objective of G is replaced by an approximation of Earth-Mover (EM) distance or Wasserstein-1. Among several other advantages, with WGANs, the phenomenon of mode collapse can be largely reduced.

There are several approaches to improving the performance of GANs [21]. For example, we can assign a label for each training sample so that the underlying distribution can be learned in a supervised fashion [32]. In our case, such labels can be monthly-based. Chen *et al.*[17] apply conditional GANs to energy applications to generate event-based scenarios for wind and solar. However, to ensure a satisfactory performance, a large amount of training data may be required.

6.4 Direct Sampling Methods

In contrast to the generative models discussed so far, we now present two simple direct sampling methods, by which the sample sequences are directly collected from historical data without help of any generative models. Unlike generative models, these can only produce a limited number of scenarios.

The first method is *boosting*. In this approach, for a paired one-year historical data of solar and load, we first construct a circular data sequence by joining their two ends. Then, starting from the beginning of each day, we collect a block of T days to form a scenario. Another direct sampling method is to only collect non-overlapping data without boosting, which we call a *non-boosting* method. As an example, if the QoS period T is equal to 100 days, there will be 365 scenarios for boosting and 3 scenarios for non-boosting in one year.

7 NUMERICAL EVALUATION

Since our sizing method cannot be directly compared to previous sizing approaches, in this section, we compare the performance of ARMA, GMMs, GANs, and the direct sampling methods in our sizing problem. We use four years (2013-2016) of data representing paired solar generation and load for 19 homes obtained from the Pecan Street Dataport [4]. We set the unit capital cost of solar panels, π_C, to be USD 2.5/W, and the unit capital cost of storage, π_B, to be USD 460/kWh [7]. The battery model parameters, which are summarized in Table 1, are calculated by following the calibration steps in [29], and correspond to Lithium-Nickel-Manganese-Cobalt battery chemistry [2, 3]. For every simulation, the initial energy state of the battery is set to be 50% of its maximum capacity.

We consider the QoS period to be 100 days and set $\theta = 5\%$, i.e., the total amount of the unmet load should not exceed 5% of the total load during 100 consecutive days. Since we focus on sizing with one-year historical data, the experiment is implemented using leave-three-out strategy. That is, the synthetic data produced by either the generative models or the direct sampling methods are based on one-year training data, and then fed to the robust sizing framework. The resultant sizing pair is then tested on the other three years[2].

Table 1: Battery model parameters

Parameter	α_c	α_d	u_1	u_2	v_1	v_2	η_c	η_d
Value	1	1	0.053	−0.125	0	1	0.99	1.11

The value of the confidence measure γ in equation (14) is set to be 0.97. For each testing year, we treat the data sequence of solar and load as circular by joining its two ends, and then check the QoS constraint (9) for all blocks of 100 days. Thus, given a testing year with 365 days, there are 365 scenarios to check.

The generative models are built separately for solar and load. The parameters of each generative model are set as follows.

- *ARMA*: For each training year, we build an ARMA model for each month. We set $q = 1$ and the maximum value of p to be 10. The best combination of (p, q) is chosen as the one with the minimum BIC. The value of α in the equation (19) is set as 0.4 using cross-validation.
- *GMMs*: The models are built based on the description in Section 6.2. To determine the number of Gaussian components I in each GMM, we test the value of I from 1 up to 4 and choose the one that leads to the minimum BIC.
- *GANs*: We use the same parameter setup as that in [17], in which GANs are applied to generate wind and solar data. The model architecture is inspired by deep convolutional GANs [37] and WGANs [9]. To fit the structure of convolutional layers, we feed data with a 10 min resolution. We additionally assign a label to each training sample based on months for supervised learning. For each training year, the real data (i.e., input to D) are formed by normalized data collected from 19 homes in the Pecan Street dataset.

Note that generative models based on ARMA and GMMs can be fit using data only for the target consumer. In contrast, GANs generally require a large amount of data for good performance, which is more than can be obtained from the target customer alone. To give GANs a better chance, we augmented the training data used for GANs with more data (19 homes). This gives GANs an extra advantage in terms of training data input. Essentially, GANs are trying to learn a general pattern of normalized data for all homes. Once the training is completed, the synthetic traces are unnormalized for the target consumer with respect to its maximum 10 min. amount.

With the generative models we can produce as many 'future' traces as we need. A large number of scenarios is preferred so that a variety of solar and load patterns can be captured. In our experiment, we gradually increased the number of scenarios until the improved performance is negligible. Concretely, we generate 500 annual hourly traces for solar and load respectively, and thus we have 500 annual scenarios. For each annual scenario, we randomly pick a block of 100 days, so we have 500 scenarios in total. For the direct sampling methods, we generate the maximum allowed number of scenarios. That is, there are 365 scenarios for the boosting method and 3 scenarios for the non-boosting method.

[2]This is much more stringent than the analysis in Reference [27], which used a leave-one-out approach.

7.1 Comparison of Generative Models

We first compare the performance of generative models in terms of how well they can capture certain characteristics of the training data. The figures we show are based on one home in 2016. We carried out the analysis for 10 other homes, which have similar performance results; these are elided for reasons of space.

In Figs. 6 and 7, we compare synthetic hourly data with the real hourly data in the dataset for both load and solar traces. Visually, the synthetic data show similar peaks and bottoms to those of real data and while exhibiting slightly different behaviour. We can clearly see that for solar power, basically all generative models exhibit diurnality. In particular, the data generated by ARMA is observed to be the most similar to the real data. Concretely, based on Fig. 6, the average Euclidean distance between the real data point and the point generated by ARMA, GMMs, and GANs is $0.02, 0.04$, and 0.05 respectively.

In Fig. 8, we show the statistical comparison for load. Fig. 8 (1) shows the empirical cumulative distribution function (CDF) for one-year hourly data. The curves for ARMA and GMMs are visually similar to the CDF computed for real data, while the curve of GANs deviates from that of real data. In Fig. 8 (2)(3), we check the sample autocorrelation for hourly and daily data. For a time series $\{X(t)\}$, the autocorrelation with respect to lag k measures the correlation between $X(t)$ and $X(t+k)$. Note that the curve of ARMA closely resembles that of the real data for both hourly and daily data, while GANs have the highest deviation. In particular, for the daily case, the sample autocorrelation of GANs is negative at the beginning and then quickly approaches zero, which is not observed in the real data. This indicates that our GANs do not adequately capture the daily autocorrelation behaviour.

In Fig. 9 we compare the same statistics for solar. We can see that all generative models are capable of characterizing the CDF of the training data. Regarding the sample autocorrelation for hourly data, the curves of all generative models resemble that of real data, with GANs deviating a bit at the peaks. The comparison for the daily case is similar to load. Again, the sample autocorrelation of GANs is observed to approach zero quickly. The large statistical deviation from the training data indicates that GANs may under-fit the data.

7.2 Sizing with Complete Hourly Data

We first assume that the target consumer has complete hourly data for solar and load, and we wish to size for the next year. Note that the only difference between the columns in Table 3 is how the synthetic traces are generated. That is, once the synthetic traces are obtained, they are fed into the same robust sizing framework, as described in Section 4. For the non-boosting method, since there are only 3 scenarios, we simply derive 3 scenario-based sizing pairs by solving the optimization problem in Section 3.2. Hence, the robust sizing pair is then chosen as the one with the highest capital cost. In Table 3 we show the results of one typical home. The conclusion is similar for the 10 other homes that we evaluated.

In Table 3, the different methods of synthetic trace generation are compared on (a) the capital cost of the system using robust sizing and (b) the total number of the failed tests (i.e., $\sum_{t=1}^{T}\delta(t)/\sum_{t=1}^{T}D(t) > \theta$) in the three testing years (i.e., the years not listed in column 1). Note that both ARMA and boosting approaches result in zero failed tests, i.e., the sizing sufficiently satisfied the load constraint for every test scenario. However the cost of the system sized with the boosting approach is much higher. For example, for training year 2013, the cost using boosting is 12% higher than ARMA, and for the training year 2015, is 25% higher. We also notice that GMMs tend to under-size and thus generally result in a high number of failed tests. The performance of systems sized with GAN-generated scenarios performance is similarly poor. We attribute this to its under-fitting of the training data. Unsurprisingly, the non-boosting method leads to the highest number of failed tests due to the insufficient number of scenarios in the training phase.

We now consider the degree to which the different scenario-generation techniques meet the load in the testing years. For each testing year, denote the value of $\sum_{t=1}^{T}\delta(t)/\sum_{t=1}^{T}D(t)$ as the percentage of the unmet load. In Fig. 10, given one training year, we additionally show the empirical CDF of the percentage of the unmet load for all testing years with the vertical dashed line in each sub-figure representing the upper bound θ. Recall that $\theta = 5\%$ in our experiment setup. Consider the event that the percentage of the unmet load is less than or equal to θ, i.e., the QoS constraint is satisfied. For ARMA and boosting, the corresponding event probability is 1. In fact, the percentage of the unmet load for these two is generally below 2% in the testing years 2013 and 2016. For GMMs and GANs, the failed tests all come from the testing year 2015, and the corresponding probability is around 0.7. For non-boosting, the corresponding probability is low with the value around 0.4 or 0.7. This result further demonstrates that an ARMA-based synthetic trace generation approach is superior to the others that we studied.

7.3 Sizing with Aggregated Data

We now consider sizing with access only to the target customer's aggregated data. The database for selecting a similar consumer consists of 19 homes located in Austin, Texas, USA, which are randomly chosen from the Pecan Street dataset. Note that in practice a large database including consumers with different consumption patterns is preferred. Thus, given a target consumer we can find a more accurate similar consumer. We assume that the hourly solar data of the target consumer is always available (i.e., only the load trace is coarse-grain), since hourly solar data can be obtained using systems such as the System Advisor Model (SAM) [6], which collate publicly-available weather data.

We consider two levels of aggregation: monthly and daily. In the monthly case, we only have 12 data points for the target consumer, each representing the aggregated monthly load consumption. For the daily case, we have 365 data points each representing the daily consumption. Since we found ARMA to have the best sizing performance in terms of the capital cost and the number of failed tests, we use ARMA to generate synthetic data.

For both monthly and daily data, the total number of failed tests is zero in all three testing years. In Fig. 11, we use the x-axis to denote the training year, and the y-axis to denote the capital cost. For comparison, we also list the cost under the complete hourly data.

Note that with the monthly data, the corresponding costs are 2% to 14% higher than those computed with hourly data. This can be understood as the cost of having incomplete information. As the time resolution of aggregated data is increased to daily, it is interesting to find that the corresponding costs are actually 1-4% lower than with hourly data. Recall that in Step 3 in Section 5, to create synthetic

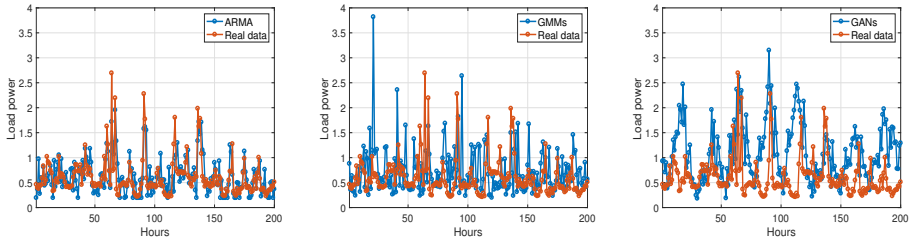

Figure 6: Comparison with synthetic load data and real load data

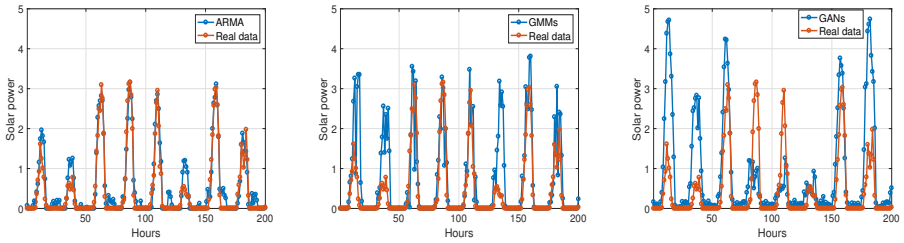

Figure 7: Comparison with synthetic solar data and real solar data

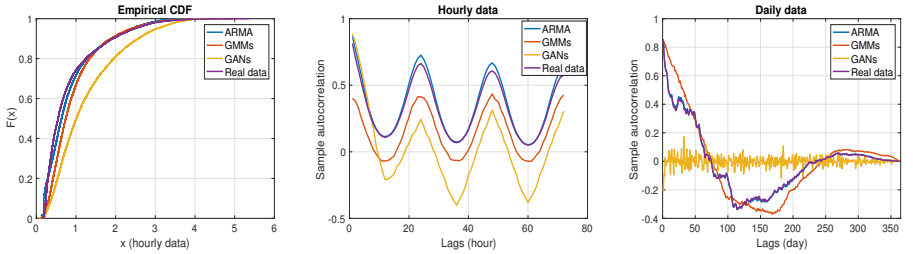

Figure 8: Comparison of generative models for load (1) empirical CDF; (2)(3) Sample autocorrelation for hourly and daily data

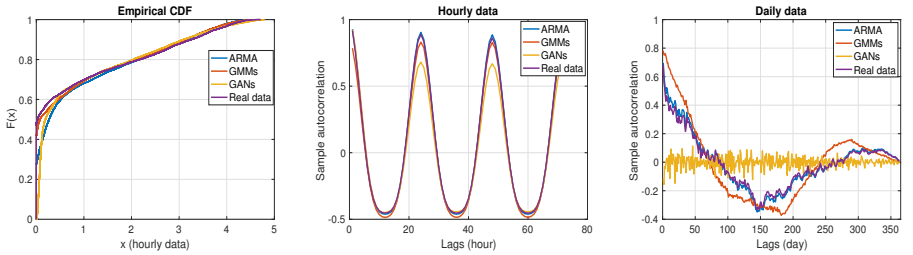

Figure 9: Comparison of generative models for solar (1) empirical CDF; (2)(3) Sample autocorrelation for hourly and daily data

data for the target consumer, we multiply the generated hourly data on the same day by the same scaling factor. Thus, the resultant adjusted synthetic data may be smoother than the actual data. This may explain a lower value of sizing in the daily case. We additionally implemented experiment on weekly and bi-weekly data, and there is no clear increasing or decreasing trend regarding the capital cost when compared to the cost with hourly data.

8 DISCUSSION AND FUTURE WORK

Our work empirically evaluated several synthetic trace generation techniques in the context of robust sizing of solar and storage. Based on prior work [17], we began our work with high hopes for GAN-based synthetic trace generation. However, in the end, we were disappointed with the approach. To begin with, GANs need to be carefully trained and the architectures of G and D can affect the algorithm performance. However, there are only very loose guidelines in making architectural decisions such as the number of hidden layers and the use of convolutional layers. However, for model-based methods such as ARMA, we can build a model based solely on data from the target consumer using well-known standard procedures.

Recall that GANs are model-free in that we do not need to specify the model parameters of the underlying data. However, a large amount of training data may be required for a satisfying performance. Although we did provide the GAN-based approach with an extra advantage by giving it access to more data, the statistics (especially

Table 2: Comparison when training with one year complete hourly data, testing on three years

	ARMA		GMMs		GANs		Boosting		Non-boosting	
Training year	Cost	Failed tests	Cost	Failed tests	Cost	Failed tests	Cost	Failed tests	Cost	Failed tests
2013	55765	0	42216	95	55472	0	62401	0	31649	352
2014	49494	0	41013	97	40464	98	51740	0	29521	514
2015	52577	0	36298	82	39998	179	65916	0	44153	0
2016	52343	0	38390	146	64021	0	57308	0	27504	770

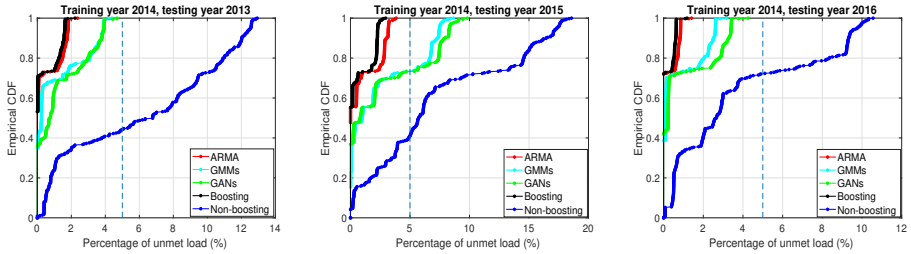

Figure 10: Empirical CDF of percentage of unmet load

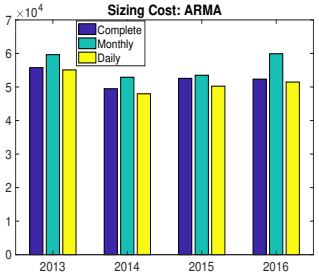

Figure 11: Comparison of sizing cost under complete, monthly, and daily data

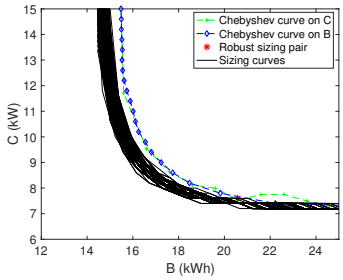

Figure 12: Generation of a robust sizing pair for $T = 365$ days: 500 sizing curves (ARMA)

for the long term) obtained from the generated samples deviate significantly from that of the historical data. We did find that GAN performance improved when the amount of input data increased. However, its performance was always worse than an ARMA-based approach in our experiments. It is possible that by using a different network architecture the performance of the GAN-based approach can be improved. However, such investigation is beyond the scope of our paper. Also, we point out that the comparison among generative models is data dependent. Based on our experiment, a model-based method such as ARMA seems to be advantageous. Comparing the generative models on other datasets is left as future work. Our expectation is that neural network-based approaches are commonly useful in more complex applications such as image generation, and classical time-series models may be good enough.

Second, in our experimental evaluation, we fixed the QoS period to be 100 days. In Fig. 2, we see that for this period, each scenario results in a distinct sizing curve. In contrast, when given a long QoS period, the variation of sizing curves diminishes. For example, in Fig. 12, for the QoS period equal to 365 days, we show the closely-spaced sizing curves and the robust sizing pair under ARMA with 500 different scenarios. In practice, the length of the QoS period is left to the consumer. For the shorter the QoS period, the more restrictive the sizing constraints. Generative models are especially useful when the QoS period is short because there may be substantial diversity of load and solar patterns for shorter durations.

Third, for the joint sizing with aggregated data, we use a Manhattan distance to measure the similarity between consumers (Eq. (15)(16)), which is linear with the difference of two points. Another common choice of distance is Euclidean distance, which is quadratic with the difference. To find a similar consumer with a smaller distance, we can enlarge the database by collecting complete data from more consumers, which is left as future work.

Finally, in this paper, we have focused on sizing with limited data (i.e., one-year historical data). Extending the proposed generative models on a different amount of historical data is left as future work.

9 CONCLUSION

We considered the robust and joint sizing of solar panels and storage using a synthetic trace-based robust sizing approach to mitigate limited historical data. We studied several trace generation models, i.e, ARMA, GMMs, GANs, as well as two direct sampling methods. The generated scenarios are then fed into a prior robust sizing framework. We found that an ARMA-based synthetic trace generation approach performed well, meeting the QoS criteria in all cases. Moreover, the greater the degree of aggregation, the more conservative the sizing. Thus, the techniques discussed in our work provide a robust foundation for sizing solar and storage systems in practical settings.

REFERENCES

[1] [n. d.]. Comparison of Different Approaches for Solar PV and Storage Sizing. ([n. d.]). https://www.dropbox.com/s/kmakva799hrrj9j/Comparison_sizing.pdf?dl=0

[2] 2016. Li-ion Polymer Cell, Kokam. (2016). li-NMC cell specifications.

[3] 2016. LIR18650 cell, EEMB. (2016). li-NMC cell specifications.

[4] 2019. Pecan Street Dataport. (2019). https://dataport.cloud/

[5] 2019. Periodogram. (2019). https://en.wikipedia.org/wiki/Periodogram

[6] 2019. System Advisor Model. (2019). https://sam.nrel.gov/

[7] 2019. Tesla. (2019). https://www.tesla.com/powerwall

[8] Shahrouz Alimohammadi and Dawei He. 2016. Multi-stage algorithm for uncertainty analysis of solar power forecasting. In *Power and Energy Society General Meeting (PESGM), 2016*. IEEE, 1–5.

[9] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*. 214–223.

[10] Rasit Ata et al. 2015. Artificial neural networks applications in wind energy systems: a review. *Renewable and Sustainable Energy Reviews* 49, 534-562 (2015).

[11] Viorel Badescu. 2014. *Modeling solar radiation at the earth's surface*. Springer.

[12] Ted KA Brekken, Alex Yokochi, Annette Von Jouanne, Zuan Z Yen, Hannes Max Hapke, and Douglas A Halamay. 2011. Optimal energy storage sizing and control for wind power applications. *IEEE Transactions on Sustainable Energy* 2, 1 (2011), 69–77.

[13] Peter J Brockwell and Richard A Davis. 2002. *Introduction to time series and forecasting*. Vol. 2. Springer.

[14] AN Celik. 2003. Assessing the suitability of wind speed probabilty distribution functions based on wind power density. *Renewable Energy* 28, 10 (2003), 1563–1574.

[15] Jairo Cervantes and Fred Choobineh. 2018. Optimal sizing of a nonutility-scale solar power system and its battery storage. *Applied Energy* 216 (2018), 105–115.

[16] SX Chen, Hoay Beng Gooi, and MingQiang Wang. 2012. Sizing of energy storage for microgrids. *IEEE Transactions on Smart Grid* 3, 1 (2012), 142–151.

[17] Yize Chen, Yishen Wang, Daniel Kirschen, and Baosen Zhang. 2018. Model-free renewable scenario generation using generative adversarial networks. *IEEE Transactions on Power Systems* 33, 3 (2018), 3265–3275.

[18] Paul Denholm, Erik Ela, Brendan Kirby, and Michael Milligan. 2010. The role of energy storage with renewable electricity generation. (2010).

[19] Ergin Erdem and Jing Shi. 2011. ARMA based approaches for forecasting the tuple of wind speed and direction. *Applied Energy* 88, 4 (2011), 1405–1414.

[20] Yashar Ghiassi-Farrokhfal, Fiodar Kazhamiaka, Catherine Rosenberg, and Srinivasan Keshav. 2015. Optimal design of solar PV farms with storage. *IEEE Transactions on Sustainable Energy* 6, 4 (2015), 1586–1593.

[21] Ian Goodfellow. 2016. NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160* (2016).

[22] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[23] Yuanxiong Guo, Miao Pan, Yuguang Fang, and Pramod P Khargonekar. 2013. Decentralized Coordination of Energy Utilization for Residential Households in the Smart Grid. *IEEE Trans. Smart Grid* 4, 3 (2013), 1341–1350.

[24] Guannan He, Qixin Chen, Chongqing Kang, Qing Xia, and Kameshwar Poolla. 2017. Cooperation of wind power and battery storage to provide frequency regulation in power markets. *IEEE Transactions on Power Systems* 32, 5 (2017), 3559–3568.

[25] Rabih A Jabr, Izudin Džafić, and Bikash C Pal. 2015. Robust optimization of storage investment on transmission networks. *IEEE Transactions on Power Systems* 30, 1 (2015), 531–539.

[26] Yuming Jiang and Yong Liu. 2008. *Stochastic network calculus*. Vol. 1. Springer.

[27] Fiodar Kazhamiaka, Yashar Ghiassi-Farrokhfal, Srinivasan Keshav, and Catherine Rosenberg. 2018. Robust and Practical Approaches for Solar PV and Storage Sizing. In *Proceedings of the Ninth International Conference on Future Energy Systems*. ACM, 146–156.

[28] Fiodar Kazhamiaka, Catherine Rosenberg, and Srinivasan Keshav. 2016. Practical strategies for storage operation in energy systems: design and evaluation. *IEEE Transactions on Sustainable Energy* 7, 4 (2016), 1602–1610.

[29] Fiodar Kazhamiaka, Catherine Rosenberg, Srinivasan Keshav, and Karl-Heinz Pettinger. 2016. Li-ion storage models for energy system optimization: the accuracy-tractability tradeoff. In *Proceedings of the Seventh International Conference on Future Energy Systems*. ACM, 17.

[30] Magnus Korpaas, Arne T Holen, and Ragne Hildrum. 2003. Operation and sizing of energy storage for wind power plants in a market system. *International Journal of Electrical Power & Energy Systems* 25, 8 (2003), 599–606.

[31] Isaac YF Lun and Joseph C Lam. 2000. A study of Weibull parameters using long-term wind observations. *Renewable energy* 20, 2 (2000), 145–153.

[32] Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[33] Juan M Morales, Roberto Minguez, and Antonio J Conejo. 2010. A methodology to generate statistically dependent wind speed scenarios. *Applied Energy* 87, 3 (2010), 843–855.

[34] Michael J Neely. 2010. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks* 3, 1 (2010), 1–211.

[35] Michiel Nijhuis, Madeleine Gibescu, and Sjef Cobben. 2017. Gaussian mixture based probabilistic load flow for LV-network planning. *IEEE Trans. Power Syst* 32, 4 (2017), 2878–2886.

[36] Anthony Papavasiliou, Shmuel S Oren, and Richard P O'Neill. 2011. Reserve requirements for wind power integration: A scenario-based stochastic programming framework. *IEEE Transactions on Power Systems* 26, 4 (2011), 2197–2206.

[37] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[38] JV Seguro and TW Lambert. 2000. Modern estimation of the parameters of the Weibull wind speed distribution for wind energy analysis. *Journal of Wind Engineering and Industrial Aerodynamics* 85, 1 (2000), 75–84.

[39] Ravindra Singh, Bikash C Pal, Rabih A Jabr, et al. 2010. Statistical representation of distribution system loads using Gaussian mixture model. *IEEE Transactions on Power Systems* 25, 1 (2010), 29–37.

[40] Bartolomeo Stellato, Bart PG Van Parys, and Paul J Goulart. 2017. Multivariate chebyshev inequality with estimated mean and variance. *The American Statistician* 71, 2 (2017), 123–127.

[41] Sun Sun, Min Dong, and Ben Liang. 2014. Real-time welfare-maximizing regulation allocation in dynamic aggregator-EVs system. *IEEE Transactions on Smart Grid* 5, 3 (2014), 1397–1409.

[42] Sun Sun, S Keshav, Catherine Rosenberg, and Matthew Peloso. 2018. Optimal Matching of Stochastic Solar Generators to Stochastic Loads. In *Proceedings of the Ninth International Conference on Future Energy Systems*. ACM, 33–37.

[43] Amit Kumar Yadav and SS Chandel. 2014. Solar radiation prediction using Artificial Neural Network techniques: A review. *Renewable and sustainable energy reviews* 33 (2014), 772–781.

[44] Chaoyue Zhao and Yongpei Guan. 2013. Unified stochastic and robust unit commitment. *IEEE Transactions on Power Systems* 28, 3 (2013), 3353–3361.

A ADDITIONAL NUMERICAL EVALUATION

In Section 7.2, we have compared the sizing results between the proposed generative models and the direct sampling methods for one consumer with a high consumption profile. In this part, we additionally include numerical results for another consumer with a low consumption profile. Again, the ARMA-based approach is observed to have the best performance in terms of the capital cost and the number of unmet QoS constraints in the testing years.

Table 3: Comparison when training with one year complete hourly data, testing on three years

	ARMA		GMMs		GANs		Boosting		Non-boosting	
Training year	Cost	Failed tests	Cost	Failed tests	Cost	Failed tests	Cost	Failed tests	Cost	Failed tests
2013	37986	0	39175	0	33799	0	40302	0	23903	426
2014	42852	0	47209	0	25709	185	44208	0	22560	631
2015	38582	0	47528	0	22999	410	40318	0	33711	0
2016	48122	0	49400	0	31690	19	48200	0	21002	549

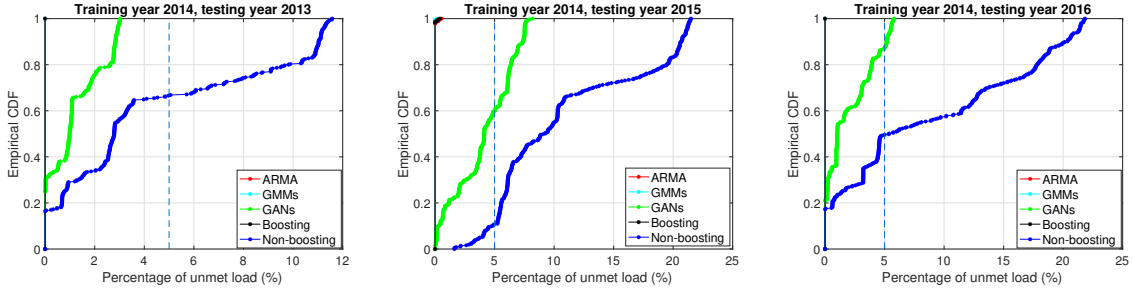

Figure 13: Empirical CDF of percentage of unmet load