Deep Generative Modeling for Probabilistic Electricity Price Forecasting

by

Nicholas Elsasser

B.S., University of Colorado Boulder, 2022

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Masters of Science

Department of Computer Science

2023

Committee Members:

Claire Monteleoni, Chair

Sriram Sankaranarayanan

Kyri Baker

Elsasser, Nicholas (M.S., Computer Science)

Deep Generative Modeling for Probabilistic Electricity Price Forecasting

Thesis directed by Prof. Claire Monteleoni

Electricity price forecasting in deregulated nodal electricity markets is a significant problem for market operators and market participants. Uncertainty in key market factors can propagate to significant price events that prove difficult to predict and hedge against. In this work, we propose and investigate a joint conditional probabilistic methodology for day-ahead price forecasting across a set of geographically distributed price nodes using recent advancements in deep generative modeling. We motivate and assess strategies that can be used to improve and adjust probabilistic forecast results and compare our proposed methodology against open access state-of-the-art statistical and machine learning benchmarks in energy price forecasting literature as well as a physics-based, commercially available optimimal economic dispatch forecasting tool.

Dedication

To all of the fluffy kitties.

Acknowledgements

Here's where you acknowledge folks who helped. But keep it short, i.e., no more than one page, as required by the Grad School Specifications.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

1.1 Dual-Settlement Nodal Electricity Markets

The increasing integration of renewable energy sources into the electrical grid has led to heightened volatility and uncertainty in electricity prices. This presents a major challenge for electricity market participants, who must be able to accurately forecast electricity prices in order to make informed investment and trading decisions. To address this challenge, the application of machine learning, and in particular deep generative models, to day-ahead electricity price forecasting has become a topic of significant interest.

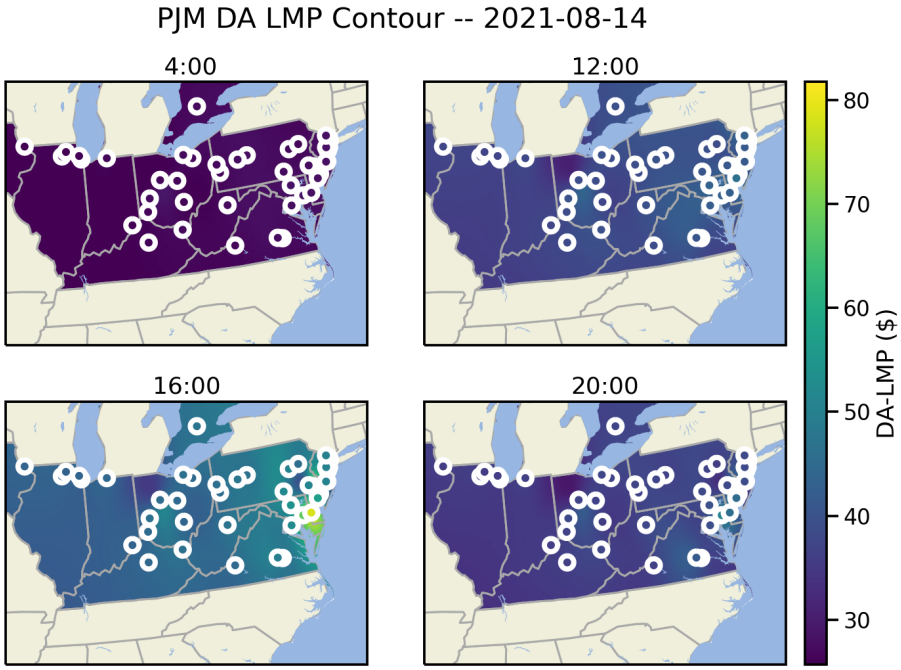
The North American electrical grid, sometimes regarded as the "world's largest machine," is a feat of modern engineering and operations that provides a critical resource for over 400 million of individuals. While the planning, construction, and maintenance of the physical infrastructure are clearly non-trivial tasks, post-completion the continuous and efficient operation of grid resources poses perhaps an even greater challenge. To keep generation and demand synchronized, a grid operator must dispatch the least-cost optimal generators such that all electrical loads *exactly* satisfied. Furthermore, the matching must be done continuously despite uncertainty in the load, weather, generation, grid topology, fuel prices, etc., lest there be a catastrophic failure in overall grid operations [3].

The early grid worked with local, vertically integrated utilities that owned all the generation and transmission in an area. Due to the small scale of the grids and relative certainty in the generation mix, real time dispatch remained simple enough such that it could be done manually.

Growth in the size and interconnection of the grid network has exacerbated the nonlinear and unpredictable interactions between components in an alternating-current (AC) network [25]. More recently, increased penetration of renewable energy sources, such as solar and wind generation, coupled with a lack of suitable storage options has dramatically increased overall uncertainty in grid operations. One proposed solution to these challenges is the deregulation of the wholesale energy market with the development of the locational marginal price (LMP) model and competitive bid-based market[CITE]. Roughly two-thirds of U.S. electricity consumers live in a region serviced by such markets, and similar efforts are actively pursued in European energy markets under the moniker "energy liberalisation."

These new, competitive energy markets allow electricity generators to offer their capacity for sale to load-serving entities, such as utilities and large energy consumers. These markets are organized by independent system operators (ISOs), which are responsible for ensuring the reliability of the electrical grid and fair markets for participants. ISOs offer multiple different markets, typically defined by the time-frame of trades, to allow participants to hedge against price volatility. One example is the day-ahead (DA) market, a forward market where electricity is traded on the day prior to delivery. Generators submit bid-curves stating how much they are able to produce and at what price, and load-serving entities (LSEs) such as utilities bid their customer's forecasted demand into the markets. Speculators play an additional role in assuming the excess risk that is unpalatable to LSEs and generators. After the trades are cleared and aggregated, the ISO formulates a locational marginal price (LMP) at locations on the grid, often referred to as *price nodes*. The LMP can be decomposed into the *energy price*, *congestion price*, and *loss price* which are the costs of producing the energy (equal at all nodes), the costs of safely transmitting energy across the grid, and the energy loss due to physical transmission inefficiencies. Figure 1.1 captures a subset of LMPs in the Pennsylvania-New Jersey-Maryland (PJM) ISO over the duration of a typical market day. Observe the uniform LMPs in the early morning hour of the market (top-left) and the significant differences between nearby prices nodes in the afternoon hours (bottom-left) indicating low-congestion and high-congestion regimes respectively.

Figure 1.1: Nodal day-ahead LMPs and the contours interpolated prices in areas between nodes. White circles depict the location of target price nodes in PJM whose prices are reflected in the inscribed colors. Prices between nodes are interpolated using radial-basis-function kernel interpolation.



1.2 The Application of Machine Learning to Energy Price Forecasting

The complicated task of forecasting nodal LMPs play a crucial role in the functioning of DA markets. Accurate forecasts allow market participants to make informed trading decisions, plan generation and demand schedules, and manage risk exposure. Thus, poor forecasts can result in reduced market efficiency, realized as wasted generation resources, high energy prices, and greater carbon emissions [**?**]. Factors such as volatile renewable generation, weather patterns, fuel prices, demand uncertainty, and complex interactions between grid components make it a formidable challenge. However, recent developments in machine learning, in particular deep learning methods, and access to historical market data show promise at capturing the complex and dynamic structures essential for accurate forecasts.

1.3 Motivation and Contribution

We motivate the development of probabilistic forecasting methods by first discussing the limitations of common point or interval forecasts. Point forecasts, despite common usage, provide the least amount of information for decision makers to act upon, as all uncertainty is discarded to obtain a single value. In comparison, interval forecasts, when paired with point forecasts, are substantially more informative. The uncertainty surrounding a point forecast can often be captured as a confidence interval (CI), i.e. a range of estimates which likely contain the "true" for some confidence level. The size of an interval can give decision makers insight to the uncertainty of the forecast as "wider" CIs indicate more uncertain forecasts. The relationship between a point forecast and interval forecast can also provide insight to the uncertainty structure. Point forecasts near the bounds of an interval forecast may indicate certain skewness and tail behavior of the true underlying uncertainty. However, interval forecasts still discard valuable structural information such as multi-modality and tail behavior outside the interval range.

Capturing all complexities and nuances of forecast uncertainty can only be achieved with probability density forecasts, whereby a probability density function is defined over the entire fore-

cast domain. Despite plenty of methods existing for low-dimensional problems, efficiently modeling joint conditional distributions in high dimensions has long been extremely difficult. Recent advances in deep neural network based generative models, in particular *normalizing flows*, show promise for characterizing high dimensional and complex joint conditional probability densities. Furthermore, normalizing flows enable both exact likelihood evaluation and efficient sampling which allow powerful Monte Carlo estimation of key quantities of interest needed to make informed financial decisions in energy markets.

The rest of this work is structured as follows. In Chapter 2 we outline the key mathematical background required to formulate probabilistic forecasts using recent advances in deep generative modeling. In Chapter 3 we define our proposed methodology for characterizing probability density estimators and generating probabilistic forecasts. In Chapter 4 we investigate techniques to improve forecasting skill in day-ahead energy markets and compare our proposed methods against open-benchmarks in energy price forecasting literature and commercial price forecasting tools. Finally, in Chapter 5 we summarize our key findings and lay out key future research paths to improve model results, robustness, interpretability, and generalizability.

1.4 Related Work

Energy price forecasting literature can be split roughly into three categories: *system pattern* (SP) methods, statistical and machine learning methods, and deep-learning methods.

The work of Zhou et al. [30] develop the idea of SPs as an exploitation in the structure of the LMP optimization problem formulation, where predictable market characteristics and associated *system pattern regions* (SPRs), or continuous neighborhoods of market conditions, represented by polytopes in load space give rise to specific predictable SPs. Forecasting price action is then reduced to estimating which SPRs future market conditions will map to, and subsequently retrieving the likely SPs and price action. To overcome bias in the set of observed market states, a "probabilistic" formulation is also presented that can report upper and lower quantile forecasts along with a mean forecast. However, the SPR state space suffers from the curse of dimensionality, making analysis

of large grids computationally intractable. Geng et al. [7] build upon the SPR concept, showing that they can be constructed using data driven methods, in particular learning SPR classification boundaries with support vector machines. These data driven SPR techniques are, however, still computationally intractable for anything more than small, synthetic grid examples. Radovanovic et al. [20] expand upon a similar idea to SPRs. Through the use of recent advances in compressed sensing they reconstruct grid topologies allowing downstream inference and clustering of congestion prices. Clusters of generation-mix and load forecast are mapped to price clusters allowing for LMP forecasting. These SPR-like representations makes this method significantly more scalable for inference, however, training the model incurs a costly step of reconstructing grid topology which can be significant for large grids [12].

Using traditional statistical methods and machine learning methods, Uniejewski et al. [21] propose a linear auto-regressive model using ordinary least squares regression and LASSO automatic feature selection techniques. This linear model can be modified for quantile regression tasks [22] or with advanced jump-diffusion and timeseries techniques to produce probabilistic forecasts [19]. Andrade et al. [1] present a methodology for both point and quantile forecasts using robust gradient boosting trees and linear quantile regression, coupled with post-processing that exploits daily average prices to increase forecast quality.

Deep-learning methods have recently risen in popularity for energy forecasting, beginning with the work of Wang et al. [24] who present a stacked de-noising auto-encoder (SDA) neural network with unsupervised pre-training and supervised fine-tuning regimes to improve forecast accuracy. Subsequent work by Lago et al. [16] compare novel deep feed-forward neural network (DNN), convolution neural network (CNN), and recurrent neural network (RNN) architectures and demonstrate the superiority of deep-learning methods over traditional statistical methods. A two stage convolutional long short term memory (CLSTM) neural network to first forecast generation bid curves to improve second stage LMP point forecasts is proposed in [28]. Zhang et al. [29] introduce a generative adversarial network (GAN) using a novel 3d tensor structure and convolutional neural network to learn the spatio-temporal interactions from the tensor representation. Cramer

et al. [4] develop a normalizing flows model capture the joint probability distribution of multi-hour price action on a single node and generate probabilistic forecasts for intra-day nodal prices in the German EPEX spot market.

Chapter 2

Mathematical Background

2.1 Measure Transport

A family of methods for density estimation fall under the umbrella term of *measure transport*. Let $\pi : \mathbb{R}^n \mapsto \mathbb{R}^+$ refer to the *target density* that we wish to estimate and $\eta : \mathbb{R}^n \mapsto \mathbb{R}^+$ refer to a known *reference density* with which we can evaluate likelihoods and draw samples from. If π and η both are smooth and continuous over $\mathbb{R}^n$ then there is a smooth bijection (diffeomorphism) $T : \mathbb{R}^n \mapsto \mathbb{R}^n$ such that the following change of variables relationship holds,

$$\pi(x) = \eta\left(z = T^{-1}(x)\right) \left|\det J_T(x)\right|^{-1},$$

where $J_T(x)$ is the Jacobian matrix of T evaluated at x. We refer to the operation $z = T^{-1}(x)$ as a *pull-back* of x in target-space to the associated z in reference-space. It is then clear that we can evaluate the likelihood of observing some x under π by a pull-back operation to find the associated z and evaluating the likelihood of z under η. Note that $\left|\det J_T(x)\right|^{-1}$ must be used to normalize changes in volume incurred by T to retain a valid probability density function; as such, it is important for T to have a tractable Jacobian determinant. Sampling $x \sim \pi$ can be accomplished by drawing a sample $z \sim \eta$ and evaluating the analogous *push-forward* operation $x = T(z)$.

Typically, the true forms of π and T must be estimated. Let $f_\theta : \mathbb{R}^n \mapsto \mathbb{R}^n$ be a diffeomorphism parameterized by θ with the resulting density estimator π_θ. The optimal parameters, θ^*, are those that minimize the difference between π_θ and π. We quantify the difference be-

tween π_θ and π using the Kullback-Leibler divergence [15], defined for two densities P and Q as $D_{\mathrm{KL}}(P||Q) = \int P(u) \log \frac{P(u)}{Q(u)} du$.

$$\begin{aligned} D_{\mathrm{KL}}(\pi||\pi_\theta) &= \int \pi(u) \log \frac{\pi(u)}{\pi_\theta(u)} du \\ &= -\int \pi(u) \log \pi_\theta(u) du + \int \pi(u) \log \pi(u) du \\ &= -\mathop{\mathbb{E}}_{u\sim\pi} [\log \pi_\theta(u)] + \mathop{\mathbb{E}}_{u\sim\pi} [\log \pi(u)] \\ &= -\mathop{\mathbb{E}}_{u\sim\pi} \left[\log \eta \left(z = f_\theta^{-1}(u)\right) - \log |\det J_{f_\theta}(u)|\right] + \mathop{\mathbb{E}}_{u\sim\pi} [\log \pi(u)] \end{aligned} \tag{2.1}$$

$\mathop{\mathbb{E}}_{u\sim\pi} [\log \pi(u)]$ is constant with respect to θ. Furthermore, if we have a collection of N samples, $\{x_i\}_{i=1}^N \sim \pi$, then we may approximate the first expected value of (2.1) with a Monte-Carlo integral,

$$\mathcal{L}_{\mathrm{MLE}}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \log \eta \left(z = f_\theta^{-1}(x_i)\right) + \log |\det J_{f_\theta}(x_i)| \,. \tag{2.2}$$

We recognize (2.2) as *maximum-likelihood loss*. Thus, minimizing the KL-divergence between π and π_θ is approximated by maximizing the likelihood of our observations under π_θ.

2.2 Normalizing Flows

Many characterizations of f_θ exist, with those using deep neural networks typically referred to as *normalizing flows*. There are many specific normalizing flow implementations. We only expand upon a variation called *discrete-time affine-coupling flows* popularized by Real-NVP [5]. For a review of current normalizing flow methods refer to [14].

The *discrete-time affine-coupling flows* constructs f_θ as the composition of L smooth and invertible functions, $f_\theta(u) = f_{\theta_L} \circ f_{\theta_{L-1}} \circ \cdots \circ f_{\theta_1}(u)$, with each $f_{\theta_\ell} : \mathbb{R}^n \mapsto \mathbb{R}^n$, $\ell \in [1, L]$ referred to as an *affine-coupling block*. Each coupling block function $v = f_{\theta_\ell}(u)$ has the following form: First splitting the components of u into two equally sized partitions $u = [u_1 \;\; u_2]$,

$$v_1 = u_1,$$

$$v_2 = u_2 \odot \exp(s(u_1)) + t(u_1).$$

Consequently, $v = [v_1 \;\; v_2]$. Functions s and t are neural networks referred to as the *scale* and *transform* networks respectively whose parameters are collectively exhaustive subsets of θ_ℓ. The structure of the affine coupling block leads to a lower-triangular Jacobian, whose log-determinant is the sum over the components from the output of s.

Affine coupling flows may limit the interactions between certain subsets of components. For instance, partitioning the input by the first and second halves for all coupling blocks would result in applying the identity to the first half of the components likely resulting in a poor density estimator. Adequate "mixing" of components can be accomplished by permuting the components between flow layers, achieved generally through random orthonormal projections or 1x1 convolutional filters, depending on the architecture of the model and application domain.

2.3 Conditional Invertible Neural Network

The conditional invertible neural network (cINN) [2] is a further development of the Real-NVP architecture that improves the expressiveness of each flow layer and add support for conditional density estimation.

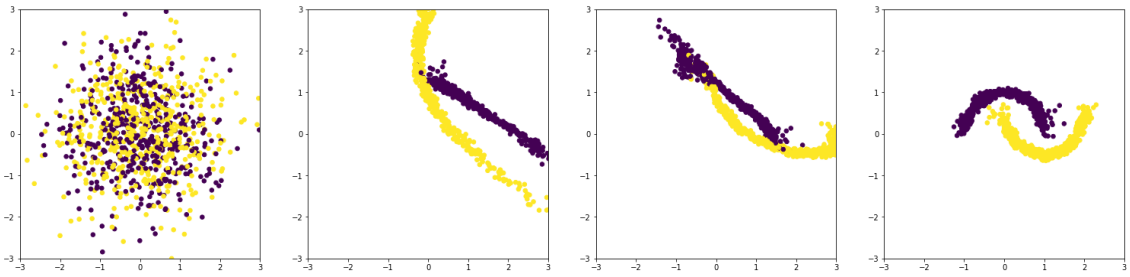
Similar to Real-NVP, we have a composition of L functions with component partitioning schemes, interspersed with $L-1$ component permutations for adequate component interaction. Let $c \in \mathbb{R}^m$ be the conditional quantities and $v = f_{\theta_\ell}(u; c)$ be a *conditional coupling block* with the following structure,

$$v_1 = u_1 \odot \exp\left(s_1(u_2; c)\right) + t_1(u_2; c) \tag{2.3}$$

$$v_2 = u_2 \odot \exp\left(s_2(v_1; c)\right) + t_2(v_1; c) \tag{2.4}$$

Functions s_1, s_2, t_1, and t_2 are neural networks whose parameters are collectively exhaustive subsets of θ_ℓ. The definition of f_{θ_ℓ} enables efficient calculation of the log Jacobian-determinant as the sum over the components of the outputs from s_1 and s_2. Including c to the input of all neural networks across all coupling blocks enables conditional density estimation, however, an unconditional density estimate can be constructed without these quantities. Following from the unconditional push-forward and pull-back operation definitions, we denote $z = f_\theta^{-1}(x; c)$ as the *conditional pull-back* operation and $x = f_\theta(z; c)$ as the *conditional push-forward* operation, whose processes are illustrated in Figure 2.1.

Figure 2.1: A conditional normalizing flow comprised of three coupling blocks applied to a synthetic dataset. **Left:** samples drawn from a reference standard Gaussian distribution. **Right:** samples after the conditional push-forward operation has been applied. **Center:** intermediate transformations as the samples are pushed through the coupling blocks.



An additional optional neural network $\tilde{c} = h(c)$ is proposed as a feature extraction model that is trained end-to-end with the normalizing flow model. When in use, the coupling block input quantities c are replaced by $\tilde{c}$. The purpose of the feature extraction network is to efficiently share informative features of the conditional quantities between all flow layers in favor of each neural network in the flow model independently extracting informative features. Inclusion of the network may be important when conditioning on large, complex quantities such as images, however, it may not be beneficial when conditioning on a small number of simple quantities such the classification of a hand-drawn digits or characters.

2.4 Generative Adversarial Networks

A generative adversarial network (GAN) [10] is a family of deep generative models characterized by two distinct models trying to out-perform one another.

The goal of a generative model is to construct a probabilistic *generator* $G : \mathbb{R}^k \mapsto \mathbb{R}^n$ such that samples drawn from $x \sim G$ are "realistic" compared to those drawn from a true target distribution $x \sim \pi$. Determining if a sample is realistic (i.e. a binary classification to determine if the sample is drawn from π) is accomplished with a secondary model referred to as the *discriminator* or critic $D : \mathbb{R}^n \mapsto [0,\ 1]$. Since G and D are unknown, let G_θ and D_ϕ be functions parameterized by θ and ϕ respectively.

The objective of the discriminator is to correctly classify whether a given observation comes from G_θ or π. Estimating ϕ by maximum likelihood yields

$$\max_{\phi} \mathop{\mathbb{E}}_{x\sim\pi} \left[\log D_\phi(x)\right] + \mathop{\mathbb{E}}_{z\sim G_\theta} \left[\log\left(1 - D_\phi(z)\right)\right]. \tag{2.5}$$

We can evaluate (2.5) via Monte Carlo integration over a set of training observations $\{x_i\}_{i=1}^N \sim \pi$ and a set of synthetic samples $\{z_i\}_{i=1}^N \sim G_\theta$. The objective of the generator is to generate samples that are indistinguishable from samples of π as judged by D_ϕ. Similarly, estimating θ by maximimum likelihood yields,

$$\max_{\theta} \mathop{\mathbb{E}}_{z\sim G_\theta} \left[\log D_\phi(z)\right]. \tag{2.6}$$

The expected value in (2.6) is denoted as the *adversarial loss* function, $\mathcal{L}_{\text{ADV}}(\theta)$, which can likewise be approximated with a collection of synthetic samples $\{z_i\}_{i=1}^N \sim G_\theta$.

While GANs are widely used for tasks requiring high dimensional and high quality samples, they suffer from two key issues that can severely limit their practicality: unstable training and the phenomenon of *mode-collapse*.

Unstable training arises from the construction of the loss for both the generator and discriminator. The loss of each model is based on the parameters of the other, thus performing parameter

updates during training results in new loss functions on every update. The changing objective landscape has the tendency to collapse or diverge during training. Significant work has gone into improving the stability of GAN training, however, achieving good results is still more art than science.

The phenomenon of *mode-collapse* is another pervasive problem of GANs, whereby the generator only learns to generate samples within a small neighborhood of the target distribution resulting in high-quality samples that lack diversity. This arises because the generator is only trained to draw samples which have high-likelihood of originating from π according to the discriminator, however, it is trivial for the generator to over-fit by generating high-quality samples only from a small region of the target distribution. An illustrative example is generating hand-drawn digits, where a *mode-covering* model would be able to generate realistic images for all digits $0-9$, a *mode-collapsed* model would only be able to generate realistic images for a subset of digits or only a single digit.

2.5 Flow-GAN

The Flow-GAN [11] model proposes using a normalizing flow for the generator function to overcome the two key issues presented with GANs by constructing a hybrid loss function for the generator combining both the *maximum-likelihood loss* and *adversarial loss* functions,

$$\min_{\theta} \mathcal{L}_{\text{MLE}}(\theta) - \lambda_{\text{ADV}} \mathcal{L}_{\text{ADV}}(\theta),$$

where $\lambda_{\text{ADV}} \geq 0$ is a hyperparameter used to blend the losses. Setting $\lambda_{\text{ADV}} = 0$ is equivalent to training a vanilla normalizing flows model ignoring the discriminator, while setting λ_{ADV} to a large number is equivalent to training a vanilla GAN.

Inclusion of $\mathcal{L}_{\text{MLE}}$ alleviates both unstable training and mode-collapse issues. First, the optimization landscape of $\mathcal{L}_{\text{MLE}}$ constant with respect to the parameters ϕ and θ, thus convergence of the generator with respect to how well it captures the target distribution from our training observations can be monitored throughout training. Second, maximizing the likelihood of observing

all training examples means mode-collapse is unlikely as any collapse during training would be heavily penalized with extremely low likelihood evaluations.

Chapter 3

Methodology

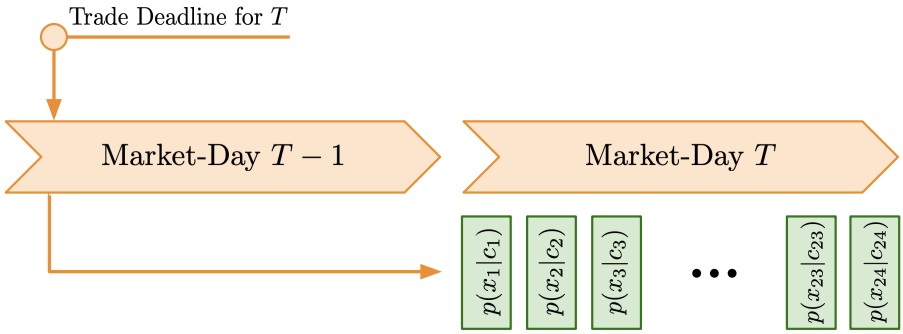
3.1 Day-Ahead Forecasting Strategy

In DA nodal markets, energy prices are settled on an hourly basis, with 24 hours per market-day. Participants are required to submit all trades — a single trade consists of a *location*, *hour*, *direction* (buy or sell), *price*, and *volume* — for a given market-day no later than one-day in advance. Specifically, if the market-day is denoted by T, there is a trade-submission deadline on the day $T-1$ at which all trades for T are submitted. For each node, prices must be forecasted for all 24 hours simultaneously. Thus, for a set of n price nodes a total of $n \times 24$ total price forecasts are required. There is no consensus on the best forecasting methodology, with popular choices being a single model creating a joint $n \times 24$ forecast [29, 28], one model per-node with 24-hour joint forecasts [1, 24, 16], one model with joint forecasts over the nodes independently per-hour [30, 7, 20], and finally one model per-hour per-node[21].

In this work, the DA price forecasting problem is approached using a single model to jointly forecast price densities across all n nodes and forecasts for each hour are generated independently. This joint probabilistic forecasting strategy captures the complex non-linear price distribution between nodes for each hour given fundamental market forecasts. That is, if $\mathcal{X}_t \subseteq \mathbb{R}^n$ is a random variable with realizations x_t of the nodal LMPs for a specific market-hour t and $c_t \in \mathbb{R}^m$ is a real-valued vector of fundamental market forecasts for the same market-hour, we propose a model which characterizes the joint conditional probability distribution $p_{\mathcal{X}_t}(x_t|c_t)$. Thus when forecasting an entire market-day, 24 independent probability densities are characterized for each hour as shown

in Figure 3.1.

Figure 3.1: Proposed day-ahead forecasting methodology. At the time of the trade submission deadline 24 joint conditional distributions have been characterized for each hour of the upcoming market-day.



The proposed strategy to characterize the joint conditional density estimators is as follows,

(1) Select a set of n price nodes for which to generate forecasts and a set of m quantities that affect day-ahead nodal energy prices and are available before the trade submission deadline.

(2) Choose an n-dimensional probability distribution to be the reference distribution η. Typical choices are standard uniform or Gaussian distributions, however any distribution with tractable log-likelihood evaluations and sampling capabilities could be used.

(3) Construct a diffeomorphism $f_\theta : \mathbb{R}^n \mapsto \mathbb{R}^n$ characterized by the cINN architecture described in Section 2.3. Additionally we must choose a suitable feature extraction network $h(c_t)$. Omission of the feature extraction network is equivalent to $h(c_t) = \mathbf{I}c_t$ where $\mathbf{I}$ is the $m \times m$ identity matrix. Note that the feature extraction network and its respective parameters are included when referring to f_θ.

(4) Construct a discriminator $D_\phi : \mathbb{R}^{n+m} \mapsto [0,\ 1]$ with which to evaluate $\mathcal{L}_{\text{ADV}}$, and select a value for λ_{ADV} to control the emphasis on $\mathcal{L}_{\text{MLE}}$ or $\mathcal{L}_{\text{ADV}}$ in the hybrid loss function.

(5) Update the parameters of f_θ and D_ϕ using the Adam stochastic gradient descent algorithm [13]. Mini-batches of pairs from the set of N historical training observations

$\{(x_i,\ c_i)\}_{i=1}^{N}$ are used to evaluate $\mathcal{L}_{\text{MLE}}$. Samples $\{z_i\}_{i=1}^{N} \sim \eta$ are drawn and the conditional push forward operation is applied to form synthetic price samples $\hat{x}_i = f_\theta(z_i; c_i)$. Mini-batches of pairs from the resulting set $\{(\hat{x}_i,\ c_i)\}_{i=1}^{N}$ are used to evaluate $\mathcal{L}_{\text{ADV}}$. Note the conditional quantities are identical in both the real and synthetic mini-batches.

3.2 Datasets

- **Prices**: Historical day-ahead LMP timeseries are obtained for 45 price nodes in the PJM market capturing prices in all 25 load zones and the interconnections with neighboring markets. These prices are published by PJM on an hourly basis shortly after the trade submission deadline for the upcoming market-day has passed. The full list of nodes can be found in Appendix A.

- **Load Forecasts**: Historical hourly load forecasts are obtained for each of the 25 load zones in PJM. Zonal load forecasts represent the expected energy demand in mega-watts (MW) in each sub-region of PJM for a given market-hour and are published on an hourly basis by PJM. These forecasts are available for participants to use during trade generation and submission. The full list of load zones can be found in Appendix B.

- **Wind Generation Forecasts**: A historical hourly timeseries is obtained for the ISO level wind generation near-term forecast. This forecast is published by PJM and represents the expected total generation (MW) from all wind turbine renewable power plants in PJM for a given market-hour and is available to participants during trade generation and submission.

- **Meteorological Forecasts**: Historical hourly meteorological forecasts are obtained for a collection of NOAA weather stations in and around PJM. There are a total of 52 weather stations spanning the entire geographical region of PJM. Each weather station reports expected hourly wind speeds (mph), temperature (°F), and dew-point (°F). These forecasts are available to participants during trade generation and submission. The full list of NOAA weather stations can be found in Appendix C.

- **Natural Gas Prices**: Historical natural gas spot price timeseries are obtained for the TETCO-M3 hub. The prices reflect the daily volume-weighted-average-price (VWAP) and are reported on a daily basis. The daily VWAP for the upcoming market-day is unknown, so instead the most recent daily VWAP at the time of trade-generation is persisted and used for all 24 hours in the price forecast.

Timeseries for all of the above datasets are captured for the entirety of years 2019, 2020, and 2021.

3.3 Data Pre-Processing

To improve the numerical stability of the stochastic gradient descent algorithm we preprocess all aforementioned quantities to normalize the disparate scaling between feature classes. A standard linear scaling strategy on a per-quantity basis was found to be sufficient when applied to all quantities despite more complex variance stabilizing scaling strategies proposed in energy forecasting literature [23]. Let $\{y_i\}_{i=1}^{N}$ be a historical timeseries for a scalar quantity such that the first $k < N$ values are used for model training and the remaining values are held out for validation, then the transformation $\bar{y}_i = \frac{y_i - \hat{\mu}_y}{\hat{\sigma}_y}$ is applied for $1 \leq i \leq N$ where $\hat{\mu}_y,\ \hat{\sigma}_y$ are the empirical mean and standard deviation of training set for the respective quantity.

In addition to the scaled fundamental market forecasts, temporal encodings of the upcoming market-time t are constructed. Assuming t contains information on the hour-of-day and elapsed-days-since-epoch (1-1-1970), given by $\text{hour}(t) \in \{1, 2, \ldots, 24\}$ and $\text{day}(t) \in \mathbb{N}$ respectively, we construct the following three sinusoidal encodings to relate temporally-similar market-times,

$$\text{Hour-of-Day} = \left[\cos\left(\text{hour}(t) \times \frac{2\pi}{24}\right),\ \sin\left(\text{hour}(t) \times \frac{2\pi}{24}\right)\right]$$
$$\text{Day-of-Week} = \left[\cos\left(\text{day}(t) \times \frac{2\pi}{7}\right),\ \sin\left(\text{day}(t) \times \frac{2\pi}{7}\right)\right]$$
$$\text{Day-of-Year} = \left[\cos\left(\text{day}(t) \times \frac{2\pi}{365}\right),\ \sin\left(\text{day}(t) \times \frac{2\pi}{365}\right)\right]$$

Note that we are targeting forecasts on 45 price nodes, thus the reference and target distributions are 45-dimensional, however, the affine coupling block architecture requires an even number of dimensions for the component partitions to be of equal sizes. To remedy this, a dummy dimension is created by appending zeros to the price-vector observations bringing the number of dimensions up to 46. When sampling price forecasts, values from dummy dimension are dropped after applying the push-forward operation.

3.4 Baselines

To validate the results of our proposed forecasting methodology we compare against the following three point-forecast baselines. Note that the first two baselines come from the open-source `epf-toolbox` [17] python library, however, the models provided in this toolbox are meant to be trained on the datasets provided by this toolbox. The dataset available for PJM only includes day-ahead prices for three price nodes and load forecasts for a single load zone and the total system-wide load. As such, the models are modified to use all available exogenous quantities and the proposed data pre-processing methodologies, however, the proposed sinusoidal temporal encoding are omitted in favor of the temporal encodings originally defined in the `epf-toolbox`.

3.4.1 LASSO Estimated Auto-Regressive

The *LASSO estimated auto-regressive model* is a parameter-rich auto-regressive model with the capacity to handle exogenous variables. Let $d(t)$ and $h(t)$ be shorthand for the functions $\text{day}(t)$, $\text{hour}(t)$ respectively for a market-time t, $p_{d(t)} \in \mathbb{R}^{24}$ be the realized day-ahead price at a single price node at market-times on the same day as $d(t)$, $x^j_{d(t)} \in \mathbb{R}^{24}$ be the j^{th} exogenous quantity at market-times on the same market-day as $d(t)$, and $\mathbf{e}_j$ the j^{th} column of the 7×7 identity matrix. Then the forecast $\hat{p}_t$ for an upcoming market-time t at a single node is defined as,

$$\begin{aligned}\hat{p}_t(\theta) = &\sum_{k\in\{1,2,3,7\}} {\theta^0_{[24\times(k-1):24\times k]}}^{\mathsf{T}} p_{d(t)-k} \\ &+ \sum_{j=1}^{m} {\theta^j_{[0:24]}}^{\mathsf{T}} x^j_{d(t)} + {\theta^j_{[24:48]}}^{\mathsf{T}} x^j_{d(t)-1} + {\theta^j_{[48:96]}}^{\mathsf{T}} x^j_{d(t)-7} \\ &+ {\theta^{m+1}}^{\mathsf{T}} \mathbf{e}_{[d(t) \mod 7]} \\ &+ \epsilon.\end{aligned}$$

Where $\theta^k \in \mathbb{R}^r$ are parameters with transpose $\theta^{\mathsf{T}} \in \mathbb{R}^{1\times r}$ and parameters $\theta^k_{[i:j]}$ are a subset of parameters θ^k characterized by the entries spanning the interval $[i, j)$ with $0 < i < j \leq r$, and $\theta^{\mathsf{T}} = [{\theta^0}^{\mathsf{T}} \dots {\theta^{m+1}}^{\mathsf{T}}]$ is the full set of parameters for the LEAR model. In this definition, θ^0 are auto-regressive parameters, $\theta^1 \dots \theta^m$ are parameters for exogenous variables, and θ^{m+1} are parameters for a day-of-the-week temporal one-hot encoding. ϵ is a standard error term.

The LEAR model is trained using sum-of-squares loss between the forecasted price $\hat{p}_t$ and the realized price p_t applying a ℓ_1 weight penalty with intensity $\lambda \geq 0$ on parameters θ to encourage sparsity. Note, θ is optimized only for a single hour, thus when training only observations at a specific hour are considered. Over a set of N training days, θ_h is optimized for a specific market hour h as,

$$\min_{\theta_h} \sum_{d=1}^{N} \left(\hat{p}_{t=(d,h)}(\theta_h) - p_{t=(d,h)}\right)^2 + \lambda\|\theta_h\|_1.$$

3.4.2 Deep Neural Network

The *deep neural network* (DNN) model is a simple feed-forward neural network which produces point forecasts for a single node at all 24 market-hours simultaneously. The DNN is trained on the same quantities as the LEAR model, that is for the upcoming market-day d, the neural network forecasts the price vector $\hat{p}_d$ using price history $\{p_{d-1}, p_{d-2}, p_{d-3}, p_{d-7}\}$, exogenous forecasts and history $\left\{x^j_d, x^j_{d-1}, x^j_{d-7}\right\}$ for exogenous quantities $1 \leq j \leq m$, and temporal encoding $\mathbf{e}_{[d \mod 7]}$.

The DNN is trained using root-mean-squared (RMS) loss, and a full description of hyperparameters can be found in Appendix D.

3.4.3 Commercial Optimal Powerflow Solver

The commercial optimal powerflow solver (OPF) is a tool that produces joint point LMP forecasts over a set of nodes for a single hour by solving a variation of the *security constrained economic dispatch* (SCED) optimization problem. The process by which LMPs are actually produced is derived from a solution of this problem, and the OPF tool aims to forecast the parameters of the SCED problem such that its solution yields price forecasts. Explanation of SCED and related power system optimization problems are beyond the scope of this work, however a review of these problems and the tools used to solve them can be found in [31]. Forecasts from commercial OPF tools are often highly desirable for their accuracy and because they are derived from the true physical properties underpinning grid and market behavior.

3.5 Evaluation Metrics

Root-mean-squared-error (RMSE), mean-absolute-percentage-error (MAPE) and mean-absolute-error (MAE) are common metrics in energy price forecasting literature, however, they are only valid for use with point forecasts. Of those metrics, MAE is the best metric for energy price forecasts because the linear errors best represent the actual losses one would incur when trading — that is the profit or loss incurred on a trade scales linearly with the difference between the buy and sell prices, analogous to the absolute error between realized and forecasted prices. Using these metrics for our forecasts would require distillation of the information-rich probabilistic forecasts down into point forecasts. Instead, we use the *continuous ranked probability score* (CRPS) [9], which generalizes the absolute error metric to probabilistic forecasts. Let $\mathbb{1}_{[x \in A]}$ be an indicator which takes the value 1 when $x \in A$ and 0 otherwise. For a univariate random variable $\mathcal{X} \subseteq \mathbb{R}$ with the probabilistic forecast represented by the cumulative density function $F_{\mathcal{X}}$, the CRPS of $F_{\mathcal{X}}$ against a realization $y \in \mathbb{R}$ is defined as

$$\text{CRPS}_{F_{\mathcal{X}}}(y) = \int_{-\infty}^{\infty} \left(F_{\mathcal{X}}(u) - \mathbb{1}_{[y \leq u]}\right)^2 du. \tag{3.1}$$

We may interpret (3.1) as the mean squared error between the forecast c.d.f. and the c.d.f. given by a Dirac delta function at the realized value. The CRPS generalizes of the absolute error of a point forecast, illustrated by the substitution of $\mathbb{1}_{[x \leq u]}$ for $F_{\mathcal{X}}(u)$ given a point forecast x,

$$\int_{-\infty}^{\infty} \left(\mathbb{1}_{[x \leq u]} - \mathbb{1}_{[y \leq u]}\right)^2 du = \int_{\min(x,y)}^{\max(x,y)} du = |x - y|,$$

thus enabling direct comparison of probabilistic and point forecasting methods.

In practice, we do not have access to the marginal c.d.f.s of each price node. Instead, we draw a set of M samples from our joint conditional density estimator and compute the empirical marginal c.d.f of the j^{th} price node, $\hat{F}_{\mathcal{X}^j}$, as

$$\hat{F}_{\mathcal{X}^j}(u) = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}_{\left[x_i^j \leq u\right]}. \tag{3.2}$$

From which we estimate the CRPS on the j^{th} price node, $\widehat{\text{CRPS}}_{\hat{F}_{\mathcal{X}}^j}$, by the substitution of (3.2) into (3.1)[1].

As this CRPS metric is only defined for univariate distributions, univariate forecasts for each price node are obtained by marginalizing samples drawn from the entire joint distribution. The average CRPS score over all price nodes for a single hour t, denoted here as mCRPS, is taken as a single distilled metric directly comparable to the MAE of a point forecast at hour t averaged over the price nodes.

$$\widehat{\text{mCRPS}}_{\mathcal{X}_t}(y) = \frac{1}{n} \sum_{j=1}^{n} \widehat{\text{CRPS}}_{\hat{F}_{x_t^j}}(y^j) \tag{3.3}$$

Additionally, the median and 99%-ile *negative log-likelihood* (NLL) metrics are evaluated on the validation dataset. A natural question is why use the median and 99%-ile NLL scores when

[1] Note that this form of the CRPS estimator is a biased estimator. For a description of an un-biased counterpart and comparisons between various estimators refer to [27]. For the remainder of this paper we continue to reference the estimator characterized by the substitution of (3.2) into (3.1) as we believe the relevant behavior of the estimator is clearest in this form.

the typical metric is the mean NLL (equivalent to $\mathcal{L}_{\text{MLE}}$). We found that despite being a valuable metric for training and parameter updates, tracking and evaluating mean NLL was unstable on our dataset due to outlier sensitivity. We would often experience a small set, or even a single validation example, that would have incredibly large NLL values causing the mean calculations to be biased by the outliers leading to unstable results making fair model comparisons difficult. Tracking median NLL values was found to be better for model comparisons as the metric is robust to outlier examples. The inclusion of the 99%-ile NLL proves useful as a large 99%-ile error imply the model is poorly fit beyond a handful of outlier observations. While certain regularization techniques we found to help reduce this behavior, we still consider it an open problem for future research to analyze model sensitivity and determine methods to improve robustness against these outlier evaluations.

Chapter 4

Experiments

We use the same cINN normalizing flow architecture, spectral-normalized GAN (SN-GAN) [18] discriminator architecture, and Adam optimizer parameters between all experiments. The details of the common model and optimizer hyperparameters are listed in Appendix D.

We split the dataset into 2019 and 2020 solely for training and hold out 2021 for evaluation. We follow a strict backtesting procedure where a model producing forecasts for an upcoming market-day has only been trained on data that would be available to a real participant prior to the market-day.

All models in this work were trained on an NVIDIA GTX 1060–6GB GPU using Python v3.10.8 and PyTorch v1.13.0. While GPU training allowed larger batch sizes and accelerated training times, often under 10 minutes, these models can be trained on CPU with smaller batch sizes, often taking between 10-30 minutes (tested on an AMD Ryzen 3700x @ 3.6GHz with 16GB RAM and a 2020 Intel i7 @ 2.3GHz 13-inch Apple Macbook-Pro with 16GB RAM).

4.1 Capturing Non-Stationary Prices

We first investigate the ability of our model to capture how energy price distributions change over time. All experiments in this section use a vanilla normalizing flows model, ignoring the adversarial loss term by setting $\lambda_{\text{ADV}} = 0$ in (2.5). As energy prices are fundamentally driven, we can ascribe changes in the price distributions to changes in factors such as *grid topology*, *fuel prices*, *population and load patterns*, *generator outages*, *renewable penetration*, *generation mix*,

weather patterns, etc. We note that a much of the variation in energy prices can be explained by changes in the fundamentals of our dataset, for example the relationship between energy prices and gas prices as shown in Figure. 4.1, seasonal load patterns as shown in Figure 4.2, or seasonal generation mixes shown in Figure 4.3.

While a well calibrated model can capture the complex, nonlinear relationships of these quantities with the powerful capacities provided by deep learning methods, these interactions change over time and with respect to data we do not have access to, such as grid topology.

As is common in energy price forecasting literature, and timeseries forecasting literature in general, we first investigate the benefits of *recalibration*, which is simply adjusting the parameters of the model as more recent information becomes available. We define our recalibration process as retraining our model from scratch every T market days where $T \in \mathbb{N}^+$ is a hyperparameter. Setting $T = 1$ is equivalent to re-training the model from scratch each day before trade generation and submission and setting $T \geq 365$ is equivalent to never recalibrating the model. For market-days between recalibration dates the model parameters stay the same and are not updated when new information comes in.

We sweep over multiple values of $T < 365$ and compare against a baseline "fixed" model that is never recalibrated over the 2021 validation set. Unsurprisingly, our results, shown in Table 4.1, make it clear that recalibrating model parameters more often produces better results. From these results, we recommend recalibrating the model as often as possible, i.e. daily. For the remainder of the experiments in this work we fix the recalibration period to 14 days as we observed this value to be good tradeoff between performance and computation time when running experiment back-tests.

Table 4.1: Comparison of various model recalibration intervals.

Recalibration Interval (T)	Median NLL (-)	99% NLL (-)	mCRPS (-)
≥ 365	-26.337	86.574	10.392
90	-41.724	65.773	9.413
30	**-48.487**	73.809	8.463
14	-44.127	**63.294**	**7.508**

Figure 4.1: Relationship between day-ahead LMPs and natural gas prices. The left-y-axis and histograms show the distribution of Chicago Hub DA-LMPs aggregated on a quarterly basis. The right-y-axis and black dots-and-flyers show quaterly aggregate natural gas prices at TETCO-M3 where the dots represent the mean price and the upper, lower flyers represent the 75^{th} and 25^{th} percentiles respectively. Observe the relationship between the mean and tails of the gas price distributions and the shape of the DA price distributions over time.

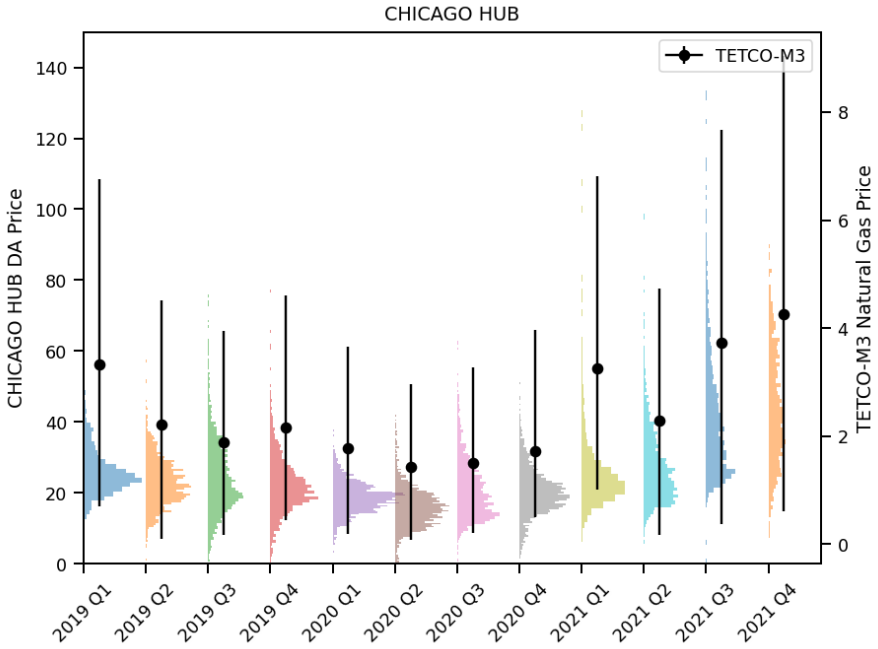


Figure 4.2: The relationship between day-ahead LMPs and seasonal load patters. The left-y-axis and blue curve shows mean total load forecast in PJM aggregated on a weekly basis. The right-y-axis and orange curve show the weekly aggregate mean Chicago Hub DA-LMP.

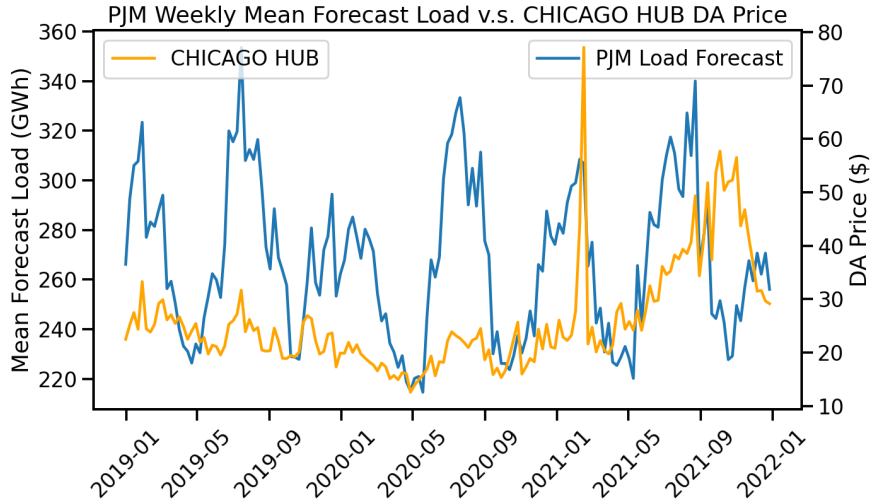
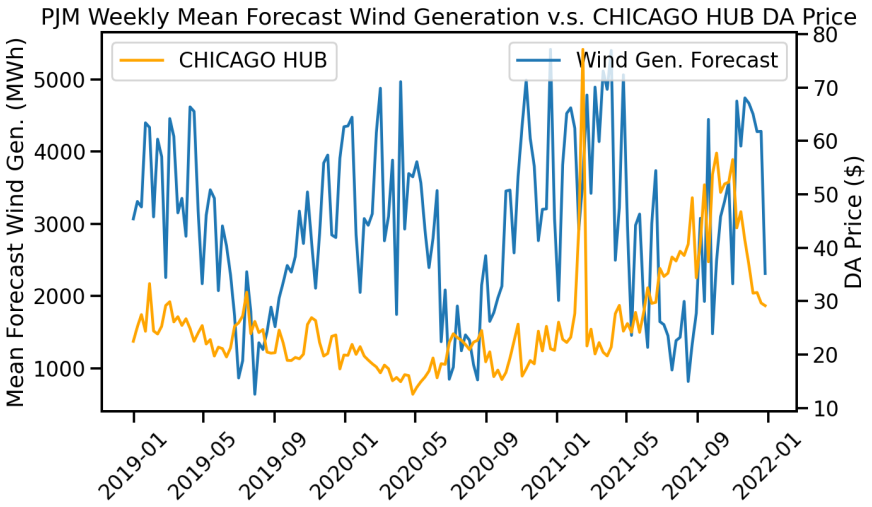


Figure 4.3: The relationship between day-ahead LMPs and seasonal wind generation patters. The left-y-axis and blue curve shows mean total wind generation forecast in PJM aggregated on a weekly basis. The right-y-axis and orange curve show the weekly aggregate mean Chicago Hub DA-LMP.



Beyond parameter recalibration, when dealing with non-stationary prices in energy price forecasting literature, and again timeseries forecasting literature in general, recent data is more informative to upcoming forecasts than data further back in history. For example, if forecasting energy prices for the upcoming week we can intuitively reason that data from the most recent two weeks in history is a more informative dataset compared to an arbitrary two weeks from two years ago. To capture this, practitioners often use a *rolling-window* of data, whereby only data from the most recent $R \in \mathbb{N}^+$ examples in history are used to estimate model parameters. While this technique may work well for models with few parameters, we found these methods to be generally unsuitable for our work, as shown in Table 4.2 where R represents the number of market-days in our training window. We believe this is caused by the deep learning model not having enough data to train and generalize well. Reducing the dataset significantly results in our model becoming significantly over-parameterized and more likely to over fit to the limited data in training windows.

Table 4.2: Comparison multiple sizes of rolling training windows.

Window Size (R)	Median NLL (-)	99% NLL (-)	mCRPS (-)
∞	-44.127	**63.294**	**7.508**
90	**-47.907**	283.582	8.327
30	-20.01	1004.566	13.73
14	25.551	418573.25	15.43

Instead, we *emphasize* the most recent or most important training observations for the upcoming forecast window. We note two possible ways one could go about this,

(1) Construct a set of weights $\{w_i\}_{i=1}^{K}$ such that $w_i \in [0, 1]$ and $\sum_{i=1}^{K} w_i = 1$ where K is the number of observations in the training set. Let weight w_i be the probability of drawing training observation i, then we can perform random sampling with replacement on our training set to train our model with emphasis on observations with greater weight.

(2) Train on all K observations in history. Then, construct a subset of $K' < K$ training observations and *fine-tune* the trained model parameters on the subset of data.

We pursue only the fine-tuning approach here. Fine-tuning is implemented by taking the subset of training observations to be the R most recent days in history at the time of model recalibration. After recalibrating the model parameters on all training examples, the optimized learning rate is reduced and training is continued on the subset of R most recent days in history. Following from previous results, we perform recalibration and fine-tuning of parameters every 14 days with no parameter updates between recalibrations. We sweep over multiple values of R and compare a baseline model which is only recalibrated but not fine-tuned. Our results for the proposed fine-tuning strategy are presented in Table 4.3.

Table 4.3: Comparison of fine-tuning rolling window sizes

Window Size (R)	Median NLL (-)	99% NLL (-)	mCRPS (-)
None	-44.127	**63.294**	7.508
14	-49.839	65.27	**5.723**
45	**-52.346**	66.59	6.686

We observe that fine-tuning on the most recent examples in history improves both model fit measured via median NLL and forecast accuracy measured via mCRPS. Moreover, reducing the size of the fine-tuning dataset to only the most relevant training examples (in this case more recent $\implies$ more relevant) has the largest impact on results. Future work may investigate a better selection method for the fine-tuning dataset to obtain important training observations beyond the most recent days in history (for example, weeks with similar weather and load patterns).

4.2 Improving Sampling Tasks with Adversarial Loss

We next investigate the impact of including the adversarial loss term, $\mathcal{L}_{\text{ADV}}$, in our model and how it specifically affects the sampling performance of the model. We utilize the same 14-day recalibration method as in previous experiments and also perform fine-tuning on the most recent 14-day window. We sweep over a range of values for $\lambda_{\text{ADV}} > 0$ and compare against a baseline model without adversarial loss, the results are shown in Table 4.4.

Table 4.4: Comparison of fine-tuning rolling window sizes

λ_{ADV}	Median NLL (-)	99% NLL (-)	mCRPS (-)
0	-49.839	65.27	5.723
0.1	-50.02	65.103	5.637
1	**-50.442**	65.176	5.569
10	-50.394	**62.854**	**5.309**

The inclusion of adversarial loss improves the mCRPS metric. To investigate why this may be the case we first note the sensitivity of our CRPS estimator to outliers in our samples. Consider the empirical c.d.f for a arbitrary distribution represented by a set of finite samples, if some of our samples are extreme outliers then they will carry a non-trivial mass under the tail of the distribution, and depending on the number and magnitude of the outliers this tail may over-estimate the tail of the true distribution resulting in a degraded CRPS estimation.

To illustrate this issue with an example, let $\mathcal{U} \sim \text{Gumbel}(\mu = 0, \beta = 1)$ with realizations u. We draw 500 samples to construct $\hat{F}_{\mathcal{U}}$ and compute the estimated CRPS against the realization $y = 0$, as shown in Figure 4.4, and find $\widehat{CRPS}_{\hat{F}_{\mathcal{U}}}(0) = 0.3331$. We next append a single synthetic outlier at $u = 10,000$ to the previous set of samples constructing $\hat{F}'_{\mathcal{U}}$ and compute $\widehat{CRPS}_{\hat{F}'_{\mathcal{U}}}(0) = 0.3748$. Clearly, extreme outliers have a detrimental effect on estimated CRPS metrics.

When observing the sample distribution drawn from a trained price density model extreme outliers can be readily noticed. Figure 4.5 shows 500 samples which estimate the joint density forecast between two price nodes marginalized out from our larger joint forecast. Most samples are indistinguishable from one-another and located near the origin as we may expect, however, we observe a small number of samples whose magnitudes are on the order of $\$10^7$. It is clear how these samples would severely degrade the CRPS metrics of this forecast.

To determine if adversarial loss address this problem in particular we first aim to quantify the occurrences of extreme outliers. We start with the matrix $\mathbf{X}_t \in \mathbb{R}^{r \times n}$ representing r samples for our forecast at time t over n price nodes. We compute the sample covariance matrix for $\mathbf{X_t}$

Figure 4.4: The CRPS of a standard Gumbel distribution, Gumbel($\mu = 0, \beta = 1$), estimated by 500 samples is calculated against an observation at 0. The top figure shows the emperical cdf of the samples while the curve in the bottom plot shows the squared error between the emperical cdf of our samples and the cdf formed by the true observation. The CRPS itself is the area under the bottom curve shown in grey.

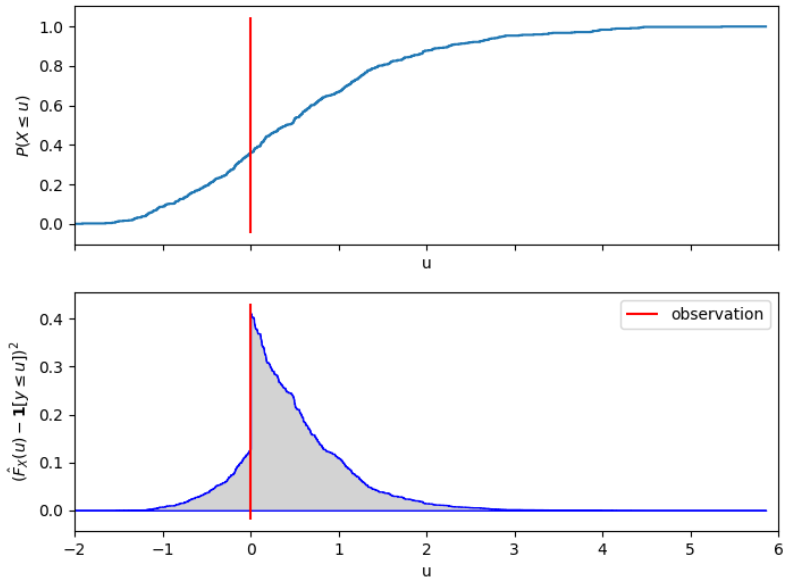
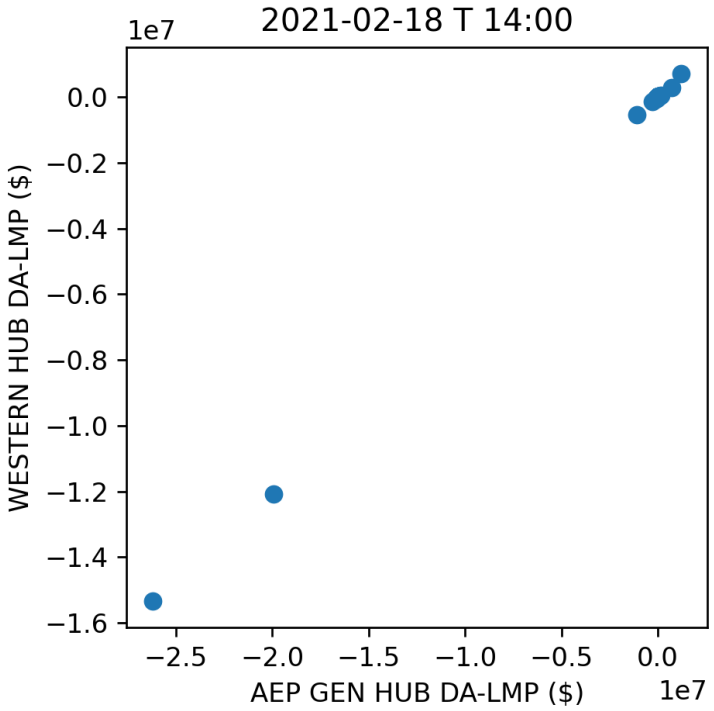


Figure 4.5: A marginalized joint distribution forecast between WESTERN HUB and AEP GEN HUB price nodes. Note the outliers with values on the order of magnitude of $\$10^7$ which exceed both reasonable price expectations and hard price limits enforced by PJM.



and subsequently the eigenvalues of the covariance matrix $\sigma_1 \dots \sigma_n$. We define the *total uncertainty* (TU) of our forcast as

$$\mathrm{TU}_{\mathbf{X_t}} = \sum_{i=1}^{n} \sqrt{\sigma_i},$$

which can be interpreted as the cumulative standard deviations in price along each orthogonal dimension in some rotated price space. We next define the *excess uncertainty* (EU) in as a binary indicator for the total uncertainty exceeding a set threshold,

$$\mathrm{EU}_{\mathbf{X}_t} = \mathbb{1}_{[TU_{\mathbf{X}_t} \geq 1000]}.$$

1000 is chosen as the threshold because day-ahead prices rarely exceed the low hundreds of dollars on more than a handful of nodes at any time, thus this threshold expected to be rarely exceeded under normal market conditions.

Counting the total excess uncertainty over samples drawn for validation forecasts then provides a metric for the frequency of outliers,

$$\text{Total Excess Uncertainty} = \sum_{t=1}^{T} EU_{\mathbf{X}_t}.$$

Comparison of the total excess uncertainty for various forecasting strategies are shown in Table 4.5. The inclusion of adversarial loss ($\lambda_{\mathrm{ADV}} = 10$) reduces the occurrences of forecasts with excessive uncertainty likely improving the CRPS metrics forecasts.

We hypothesize this effect is due to the "contraction" of the reference density when transforming to the target density under adversarial loss due to naturally arising mode-collapse phenomena. When using a normalizing flow model as the generator of a GAN we may choose to understand mode collapse as the contraction of the target density to a small neighborhood in the target-space[1]

.

[1] Indeed, results from the original FlowGAN work [11] show an exploding $\mathcal{L}_{\mathrm{MLE}}$ when trained only on $\mathcal{L}_{\mathrm{ADV}}$ despite high-quality samples still being drawn which furthers this density contraction hypothesis.

Table 4.5: The total excessive uncertainty is captured for four models over the training set. Recalibration and fine-tuning sizes are both set to 14, $\lambda_{\text{ADV}} = 10$. The model denoted as *none* has no recalibration, fine-tuning, or adversarial loss. Strategies are assumed to be ommitted unless specified in the model name.

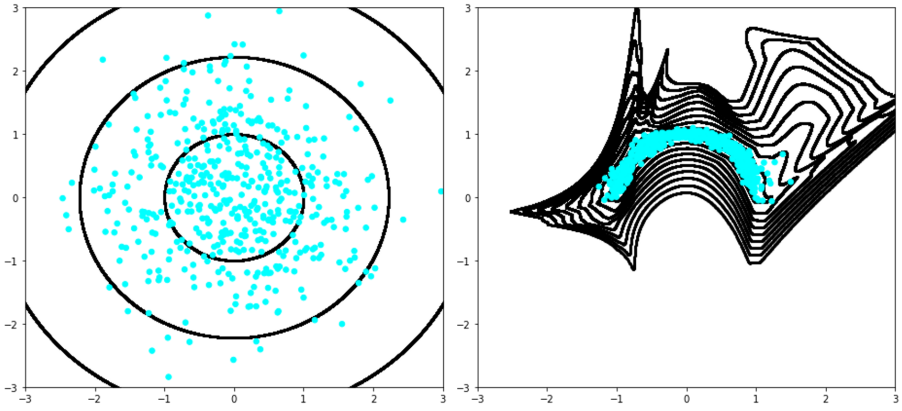
Model Strategy	Total Excess Uncertainty
None	34
Recal.	180
Recal. + Fine-tune	379
Reval. + Fine-tune + Adv. Loss	72

We believe this contracting behavior helps regularize the mapping of the normalizing flow model in the low-density tails of the estimated target density by pulling density from the tails of the distribution in towards the modes. In Figure 4.6 we show a synthetic distribution whose mapping is learned with a normalizing flow model. Observe the warping of the reference domain by a set of concentric circles drawn in the reference space and their subsequent mappings in the target space. Near the tails of the target density there is excessive warping of the reference domain as it becomes more difficult for the normalizing flow model to accurately capture low-probability tail behavior due to a lack of training observations and penalty for excessive warping of the domain. Extrapolating to higher dimensions, this warping would likely cause the occasional sample in reference space to be mapped to an egregious outlier in the target space due to a manifestation of the curse of dimensionality. Thus, the hypothesized regularizing properties of adversarial loss would reign in excessive warping under our estimated mapping. In the context of our work it is enticing to refer to the inclusion of adversarial loss as *adversarial regularization* due to the properties observed.

4.3 Comparison Against Baselines

Finally, we compare our proposed probabilistic forecasting methodology (including recalibration, fine-tuning, and adversarial regularization) against several baseline traditional point forecasting methods. We first compare against two open source state-of-the-art energy price forecasting models in literature provided by the `epf-toolbox` [17] python library. We compare against both

Figure 4.6: The warping of the reference domain under the push-forward opertion of a trained normalizing flow model with with three affine-coupling layers. **Left:** Reference samples shown in blue and concentric circles in the reference domain drawn in black. **Right:** Samples pushed-forward into the target space and the same concentric circles warped in the target domain.



the modified *lasso estimated auto-regressive* (EPF-LEAR) model and modified *deep neural network* (EPF-DNN) model as described in Section 3.4. We note that a significant downside to these models is they are only defined for single price nodes, making multi-node forecasts significantly more expensive to obtain and the resulting forecasts would likely fail sufficiently to capture and joint structure between node forecasts. As such, we only perform comparisons at a select few price nodes. For each price node, a single DNN model and 24 hourly LEAR models a constructed. Univariate density forecasts for each price are obtained by marginalizing out samples from the entire joint density forecast. To match the parameter recalibration of our proposed models, we follow the same training and recalibration procedures with both the EPF-LEAR and EPF-DNN models. Comparisons between the CRPS of our probabilistic forecasts (cINN) against the MAE of the EPF-DNN and EPF-LEAR forecasts are shown in Table 4.6. The proposed density forecasting method produces more accurate forecasts than state-of-the-art point forecasting methods. Note, beyond greater forecast accuracy our proposed methodology also captures joint structures between price nodes and uncertainty in forecasts which non-probabilistic forecasts fail to provide.

Table 4.6: Comparison of forecast accuracy between our probabilistic forecasting strategy and state-of-the-art models from the `epf-toolbox` [17] python library over single price nodes.

Price Node \ Model	EPF-LEAR (MAE)	EPF-DNN (MAE)	cINN (mCRPS)
CHICAGO HUB	6.622	6.231	**5.341**
DOMINION HUB	7.077	6.565	**6.174**
EASTERN HUB	7.095	7.526	**6.087**
NEW JERSEY HUB	4.923	5.025	**3.983**
OHIO HUB	6.444	6.119	**5.558**

Next, we compare against forecasts from the commercial OPF solver. The solver only produces point forecasts on a subset of 14 of our price nodes, thus we marginalize out the joint density forecast on those specific nodes when comparing against our probabilistic forecasts. Again, we compare the MAE of the OPF forecasts against the mCRPS of our probabilistic forecast with our results shown in Table 4.7. Our proposed probabilistic forecasting strategy is more accurate than

the commercial, physics-based price forecasting tool, again with the added benefit of information-rich joint probabilistic forecasts as opposed to low-information point forecasts.

Table 4.7: Comparison of forecast accuracy between our proposed probabilistic forecasting strategy and a commercial OPF solver over a subset of 14 prices nodes.

Model	MAE (-)	mCRPS (-)
OPF	6.666	-
cINN	-	**5.435**

Figure 4.7 illustrates forecasts from our joint density method and the three baseline methods against observed DA prices on a single node over two time intervals in 2021. Observe how the uncertainty in our proposed density forecasting method changes with respect to market conditions. Also note that beyond the mean forecast and 95% confidence intervals, each forecast is a complex probability density function as shown in Figure 4.8.

Figure 4.7: Forecasts from the proposed probabilistic forecasting methodology, EPF-LEAR, EPF-DNN, and OPF baselines against the true NEW JERSEY HUB DA LMPs over two five-day intervals in 2021. (a) forecasted and observed LMPs over five-days during an extreme winter weather event in Texas that caused unprecendented price action in PJM. (b) forecasted and observed LMPs over a typical five-day interval in the summer. Note the differences in uncertainty and intra-day price action captured by the probabilistic forecasts during the different market conditions.

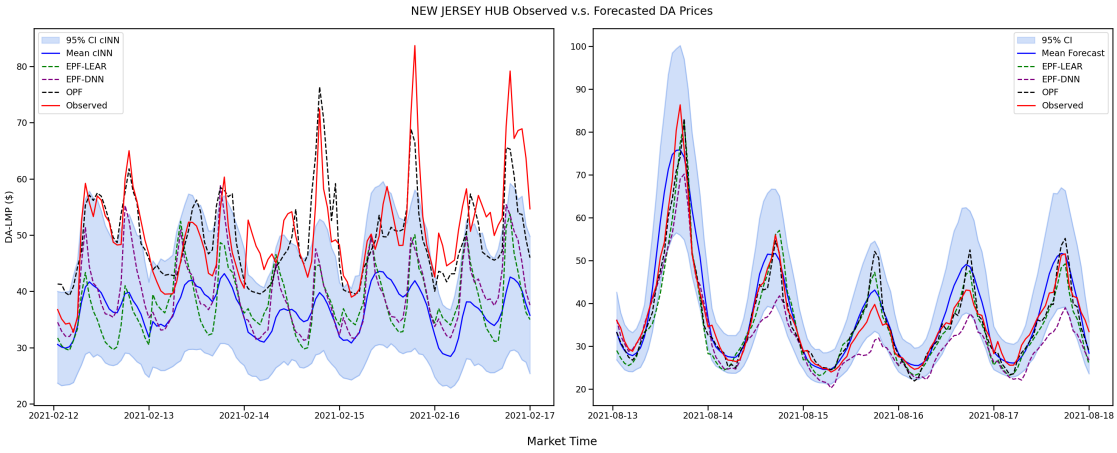
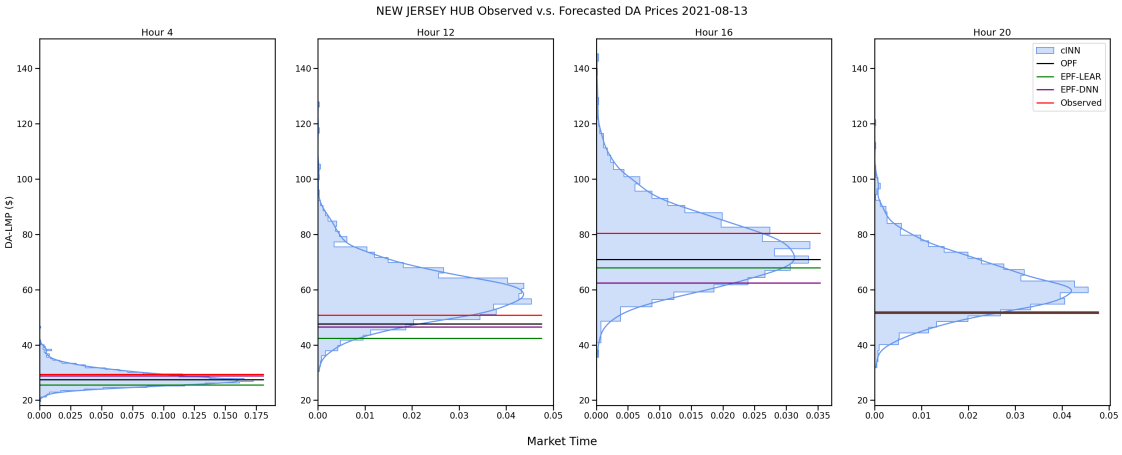


Figure 4.8: Forecasts for our proposed probabilistic forecasting methodology and EPF-LEAR, EPF-DNN, and OPF baselines against the true DA prices are shown at the NEW JERSEY HUB price node over four hours in a single market day.



Chapter 5

Conclusion

In this work we presented a novel methodology for joint probabilistic day-ahead energy price forecasts over a set of geographically distributed price nodes and several practices that can be used to improve forecasting results using recent advances in deep generative modeling. Our proposed methodology allows for both exact conditional likelihood estimation and efficient conditional sampling for forecasting tasks. Comparisons of our model to both state-of-the-art open-access benchmarks in energy price forecasting literature and commercial physics based optimal power-flow tools provide clear evidence for the superiority of our proposed methodology both in terms of forecast accuracy, and more expressive information-rich probabilistic forecasting in comparison to traditional point-forecasting methods. Despite this, we believe the methodology presented in this work can be further improved with the following lines of continued research,

(1) **Model Robustness and Interpretability**: A key issue we faced during development of these methods was the use of non-standard robust metrics when tracking model fit due to the instabilities we observed for mean NLL calculations, caused by sporadic and extremely low log-likelihood evaluations. While we were able to find workarounds, they do not address the root causes of these instabilities. We believe the application of modern robust deep learning methods to be a worthwhile investigation to determine if more stable results can be achieved by improving model robustness. Additionally, such techniques could be used to provide interpretability as to the quantities and market conditions to which our model is most sensitive. Such transparency is desirable for the utilization of these models when

executing positions in the markets based on information provided by these forecasts.

(2) **Hierarchical Modeling and Uncertainty Propagation**: A key assumption of our methods is that the quantities we condition our forecasts on have no uncertainty, however, load, generation, and meteorological forecasts are known to be uncertain and error-prone. While more accurate forecasts would likely improve results, no forecast is 100% correct, 100% confident, 100% of the time. Instead, capturing the uncertainty of those upstream fundamental forecasts in their own density estimate $p(c_t)$ and constructing a hierarchical density estimate $p(x_t|c_t)p(c_t)$ may better capture the relationship between our fundamental forecasts and resulting price forecasts as we could propagate the uncertainty in our fundamental forecasts into the uncertainty of our price forecasts.

(3) **Generalization to Other Markets and a Greater Number of Price Nodes**: Our experiments in this work were limited to the study of a single set of 45 price nodes in the PJM market. There are a total of 8 day-ahead energy markets in North America, with many more in Europe and the rest of the world. These markets each can have significant differences between one another in terms of regulation, available generation, load profiles, weather patterns, etc. that all have outstanding affect on day-ahead price actions. Additionally, we only looked a small set of 45 price nodes despite markets often being comprised of hundreds if not thousands of distinct price nodes. We believe further research into the generalizablility of our proposed methods into other markets and the issues faced when scaling up forecasts to larger sets of price nodes to be of great practical importance to the deployment and execution of this methodology.

Bibliography

[1] José Andrade, Jorge Filipe, Marisa Reis, and Ricardo Bessa. Probabilistic price forecasting for day-ahead and intraday markets: Beyond the statistical model. Sustainability, 9:1990, 10 2017.

[2] Lynton Ardizzone, Jakob Kruse, Carsten Lüth, Niels Bracher, Carsten Rother, and Ullrich Köthe. Conditional invertible neural networks for diverse image-to-image translation, 2021.

[3] POWER Committee. The timeline and events of the february 2021 texas electric grid blackouts, July 2021.

[4] Eike Cramer, Dirk Witthaut, Alexander Mitsos, and Manuel Dahmen. Multivariate probabilistic forecasting of intraday electricity prices using normalizing flows, 2022.

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2016.

[6] Jonathan Dumas, Antoine Wehenkel, Damien Lanaspeze, Bertrand Cornélusse, and Antonio Sutera. A deep generative model for probabilistic energy forecasting in power systems: normalizing flows. Applied Energy, 305:117871, 2022.

[7] Xinbo Geng and Le Xie. Learning the lmp-load coupling from data: A support vector machine based approach. IEEE Transactions on Power Systems, 32(2):1127–1138, 2017.

[8] Xavier Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. Journal of Machine Learning Research - Proceedings Track, 9:249–256, 01 2010.

[9] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. Journal of the American Statistical Association, 102(477):359–378, 2007.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[11] Aditya Grover, Manik Dhar, and Stefano Ermon. Flow-gan: Combining maximum likelihood and adversarial learning in generative models, 2017.

[12] Vassilis Kekatos, Georgios B. Giannakis, and Ross Baldick. Online energy price matrix factorization for power grid topology tracking. IEEE Transactions on Smart Grid, 7(3):1239–1248, 2016.

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[14] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(11):3964–3979, nov 2021.

[15] S. Kullback and R. A. Leibler. On Information and Sufficiency. The Annals of Mathematical Statistics, 22(1):79 – 86, 1951.

[16] Jesus Lago, Fjo De Ridder, and Bart De Schutter. Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms. Applied Energy, 221:386–405, 2018.

[17] Jesus Lago, Grzegorz Marcjasz, Bart De Schutter, and Rafał Weron. Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark. Applied Energy, 293:116983, 2021.

[18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018.

[19] Peru Muniain and Florian Ziel. Probabilistic forecasting in day-ahead electricity markets: Simulating peak and off-peak prices. International Journal of Forecasting, 36(4):1193–1210, 2020.

[20] Ana Radovanovic, Tommaso Nesti, and Bokan Chen. A holistic approach to forecasting wholesale energy market prices. IEEE Transactions on Power Systems, 34(6):4317–4328, 2019.

[21] Bartosz Uniejewski, Jakub Nowotarski, and Rafał Weron. Automated variable selection and shrinkage for day-ahead electricity price forecasting. Energies, 9(8), 2016.

[22] Bartosz Uniejewski and Rafał Weron. Regularized quantile regression averaging for probabilistic electricity price forecasting. Energy Economics, 95:105121, 2021.

[23] Bartosz Uniejewski, Rafał Weron, and Florian Ziel. Variance stabilizing transformations for electricity spot price forecasting. IEEE Transactions on Power Systems, 33(2):2219–2229, 2018.

[24] Long Wang, Zijun Zhang, and Jieqiu Chen. Short-term electricity price forecasting with stacked denoising autoencoders. IEEE Transactions on Power Systems, 32(4):2673–2681, 2017.

[25] Yang Yang, Takashi Nishikawa, and Adilson E. Motter. Small vulnerable sets determine large network cascades in power grids. Science, 358(6365):eaan3184, 2017.

[26] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning, 2017.

[27] Michaël Zamo and Philippe Naveau. Estimation of the continuous ranked probability score with limited information and applications to ensemble weather forecasts. Mathematical Geosciences, 50(2):209–234, Feb 2018.

[28] Zhongxia Zhang and Meng Wu. Energy price prediction considering generation bids variation: A two-stage convolutional long short-term memory approach. In 2022 IEEE Power Energy Society General Meeting (PESGM), pages 1–5, 2022.

[29] Zhongxia Zhang and Meng Wu. Predicting real-time locational marginal prices: A gan-based approach. IEEE Transactions on Power Systems, 37(2):1286–1296, 2022.

[30] Qun Zhou, Leigh Tesfatsion, and Chen-Ching Liu. Short-term congestion forecasting in wholesale power markets. IEEE Transactions on Power Systems, 26(4):2185–2196, 2011.

[31] Jizhong Zhu. Optimization of Power System Operation. Wiley-IEEE Press.

Appendix A

List of PJM Price Nodes

Day-ahead hourly LMPs from 2019, 2020, and 2021 were collected from the following 45 PJM nodes,

(1) AECO RESID AGG
(2) AEP GEN HUB
(3) AEP-DAYTON HUB
(4) AEPAPCO RESID AGG
(5) AEPIM RESID AGG
(6) AEPKY RESID AGG
(7) AEPOHIO RESID AGG
(8) APS RESID AGG
(9) ATSI GEN HUB
(10) BGE RESID AGG
(11) CHICAGO GEN HUB
(12) CHICAGO HUB
(13) COMED RESID AGG
(14) DAY RESID AGG
(15) DEOK RESID AGG
(16) DOMINION HUB
(17) DOM RESID AGG
(18) DPLEASTON RESID AGG
(19) DPL RESID AGG
(20) DUQ RESID AGG
(21) EASTERN HUB
(22) EKPC RESID AGG
(23) FEOHIO RESID AGG
(24) HUDSONTP
(25) IMO
(26) JCPL RESID AGG
(27) LINDENVFT
(28) METED RESID AGG
(29) MISO
(30) N ILLINOIS HUB
(31) NEPTUNE
(32) NEW JERSEY HUB
(33) NYIS
(34) OHIO HUB

(35) OVEC RESID AGG

(36) PECO RESID AGG

(37) PENELEC RESID AGG

(38) PENNPOWER RESID AGG

(39) PPL RESID AGG

(40) PSEG RESID AGG

(41) RECO RESID AGG

(42) SMECO RESID AGG

(43) UGI RESID AGG

(44) VINELAND RESID AGG

(45) WESTERN HUB

Appendix B

List of PJM Load Zones

Hourly load forecasts from 2019, 2020, and 2021 were collected from the following 25 PJM zones,

(1) AECO
(2) AEP
(3) AP
(4) ATSI
(5) BGE
(6) COMED
(7) DAYTON
(8) DEOK
(9) DOMINION
(10) DPL
(11) DUQUESNE
(12) EKPC (PJMISO)
(13) JCPL
(14) METED
(15) MID-ATLANTIC REGION
(16) PECO
(17) PENELEC
(18) PEPCO
(19) PPL
(20) PSEG
(21) RECO
(22) RTO COMBINED
(23) SOUTHERN REGION
(24) UGI
(25) WESTERN REGION

Appendix C

List of NOAA Weather Stations

Hourly wind speed, temperature, and dew-point forecasts from 2019, 2020, and 2021 were collected from the following 52 NOAA weather stations,

(1) DE - Wilmington/Airport

(2) IN - Fort Wayne/Baer Field

(3) IN - Indianapolis/Intl/NWSFO

(4) IN - Lafayette/Purdue University/ASO

(5) IN - South Bend/St. Joe

(6) IN - Terre Haute/Hulman

(7) KY - Covington/Cincinnati

(8) KY - Lexington/Bluegrass

(9) KY - Louisville/Standiford

(10) MD - Annapolis

(11) MD - Baltimore (until 1979)

(12) MD - Baltimore/Science Center

(13) MI - Ann Arbor/Municipal

(14) MI - Bad Axe/Memorial

(15) MI - Detroit/City Airport

(16) MI - Detroit/Metropolitan

(17) MI - Grand Rapids/Intl/NWSFO

(18) MI - Houghton Lake/Airport

(19) MI - Kalamazoo/Intl Arpt

(20) MI - Lansing/Capital

(21) MI - Sault Ste Marie/Sanderson

(22) MI - Traverse City/Cherry

(23) NC - Charlotte/Douglas

(24) NC - Raleigh/Durham

(25) NJ - Atlantic City/Intl

(26) NJ - Newark/Intl

(27) NJ - Trenton/Mercer County

(28) OH - Akron

(29) OH - Akron-Canton/Regional

(30) OH - Cincinnati/Lunken

(31) OH - Cleveland/Hopkins/NWSFO

(32) OH - Columbus/Port Columbus Intl

(33) OH - Dayton/James M Cox Municipal

(34) OH - Dayton/Wright-Patterson AFB

(35) OH - Toledo Express

(36) OH - Toledo/Metcalf Field

(37) PA - Allentown-Bethlehem

(38) PA - Erie/Intl

(39) PA - Harrisburg Intl

(40) PA - Philadelphia/Intl

(41) PA - Pittsburgh/Intl

(42) PA - Wilkes-Barre/Scranton

(43) PA - Williamsport

(44) VA - Norfolk/Intl Arpt

(45) VA - Richmond/Byrd Field

(46) VA - Richmond/Hanover

(47) VA - Roanoke/Municipal

(48) VA - Wallops Island/Station

(49) VA - Washington/Dulles Intl Arpt

(50) WV - Charleston/Kanawha

(51) WV - Elkins/Randolph Field

(52) WV - Huntington/Tri State

Appendix D

Model and Optimizer Hyperparameters

D.1 cINN

For all experiments we construct our cINN normalizing flow generator as follows,

Table D.1: Fixed hyperparameters of the cINN generator model.

Parameter Name	Value
Reference Distribution	Standard Multivariate Gaussian
# Flow Layers	12
Flow NN Hidden Layers	1
Flow NN Hidden Units	128
Flow NN Activation	ReLU
Feature Extraction Network	Omitted[1]
p-Dropout	0.8
Spectral Norm Regularization [26]	0.1
Weight Initialization	Uniform Xavier [8]

Additionally, we used a soft-clamping on the outputs of the s_1 and s_2 neural networks in each flow layer $\tilde{s} = \frac{2\alpha}{\pi} \arctan\left(\frac{s}{\alpha}\right)$ to prevent exploding values from exponentiation. We take $\alpha = 1.9$ as recommended [2].

[1] We experimented with many settings for a deep feed-forward neural network projecting into both higher and lower dimensional latent feature spaces, however, we found that inclusion of the feature extraction neural network tended to significantly increase NLL metrics while degrading mCRPS metrics. We believe a valuable line of future work to be further investigation of the root cause of the divergence between these metrics and methods for reducing the gap.

D.2 GAN Discriminator

For all experiments we construct a feed-forward spectral-normalized [18] GAN discriminator as follows,

Table D.2: Fixed hyperparameters of the SN-GAN discriminator model.

Parameter Name	Value
Activation	LeakyReLU($\alpha = 0.2$)
Hidden Layer %[2]	[0.8, 0.4, 0.2, 0.1]

D.3 EPF-DNN

We construct a modified version deep neural network presented in the open-access `epf-toolbox` used in the experiments of section 4.3 as follows,

Table D.3: Fixed hyperparameters of the EPF-DNN model.

Parameter Name	Value
# Hidden Layers	1
# Hidden Units / Layer	512
Activation	LeakyReLU
p-Dropout	0.4
Spectral Norm Regularization	0.01

D.4 Adam Optimizer

All deep-learning models are trained using the Adam optimizer with parameters as follows,

[2] The number of hidden units per layer is calculated as a percentage of the number of input (# of price nodes + # conditional quantities), i.e. $\texttt{hidden size}_i = \texttt{input size} \times hiddenpct_i$.

Table D.4: Fixed hyperparameters of Adam optimizers.

Parameter Name	Value
β_1	0.9
β_2	0.98
Learning Rate	10^{-3}
Fine-tune Learning Rate	10^{-4}
Batch Size	128