Distribution-Free Multi-Horizon Forecasting and Vending System

Vincent Quenneville-Bélair
Amazon
New York, USA
quennv@amazon.com

Malcolm Wolff
Amazon
Seattle, USA
wolfmalc@amazon.com

Brady Willhelme
Amazon
Seattle, USA
bwwillhe@amazon.com

Dhruv Madeka
Amazon
New York, USA
maded@amazon.com

Dean Foster
Amazon
New York, USA
foster@amazon.com

ABSTRACT

We introduce a new distribution-free multi-horizon forecast. As such, it integrates product-level forecast at fixed lead times, spans, and quantiles with the ability for a user to request a forecast at any other lead time, span or quantile via piecewise linear interpolation with exponential extrapolation for the tails of the distribution. The selected algorithm leads to a reduction in weighted quantile loss over the current methodology of over 13.4% for percentiles less than 3 and 1.9% for percentiles over 97 with measured increase of no more than 0.4% elsewhere. We also implement a new override blending algorithm allowing users to override any quantiles while maintaining consistency.

CCS CONCEPTS

• **Computing methodologies → Modeling methodologies**.

KEYWORDS

neural network,time series analysis,machine learning

Forecasting time series occur in a wide range of real-world problems such as urban planning [2], retail [6], energy [17] and tourism [1]. The goal typically consists of estimating the future values of multiple time series given past observations. We focus on the fundamental machine learning problem of time series forecasting as applied to supply chain management of a large e-commerce retailer. This application requires the predictions of many correlated time series over multiple horizons. The lead time and span of the forecast describe the horizon of interest: they are the start and duration of the planning period, respectively. That is, a forecast for a lead time of three weeks and span of one week may be used to purchase a product that takes three weeks to arrive at a fulfillment center, with orders placed once a week.

Beyond point estimation, the supply chain management decisions to be made requires a measure of uncertainty that distributional forecasts provide. The quantile of interest depends on the application in question. For example, when we are considering purchasing an product for the retail market, we are typically interested in forecast values at or above the 50th quantile of the distribution. In contrast, when we are interested in marking down the price of a product that is not selling as expected, forecast values at or below the 10th quantile are of greater interest.

Motivated by this real-world business context, we thus generate a demand forecast via quantile regression wherein two quantiles (the 50th and 90th) are predicted by a deep-learning model. In retail inventory management, all quantiles do not have the same business relevance. As such, it is sensible to use the 50th percentile as a guardrail to ensure that we buy at least a certain amount of an item, and the 90th to have enough in stock to cover demand most of the time. Neural networks have led to improvements in time series forecast accuracy; starting with MQ-RNN and MQ-CNN in Wen et al. [21] and, more recently, an implementation of a Transformer architecture [18] in MQ-Transformer [5]. These models generate forecasts for a particular product based on its demand history, product attributes and known future information (e.g. promotions, planned sales, holidays).

We then fit a truncated and shifted gamma distribution (TSGD) using the two aforementioned quantiles in order to obtain the mean, variance and other quantiles of the distribution. The gamma distribution is a two-parameter probability distribution, denoted $\text{Gamma}(k, \theta)$ with shape k and scale θ, which is skewed to the right as we expect from demand. We shift the distribution to the left by a constant value of -1 to model the probability of a product having no demand in a given period, since $x = 0$ is not in the support of this distribution. This in turn yields non-zero probability for negative demand values, so the distribution is truncated at $x = 0$ [20]. With the TSGD, we can generate forecast values for all quantiles at predetermined forecast horizons. The collection of these forecast horizons is referred to as a "mesh". We can then extend this to forecast at any lead time and span within the bounds of that mesh, but outside the predetermined horizons. We use barycentric interpolation of the mean and variance to generate quantiles at those forecast horizons, because quantiles are not linear quantities like the mean. Here, we propose a replacement for the fitting of the TSGD and barycentric interpolation in order to more accurately capture quantiles beyond the 50th and 90th quantiles at any forecast horizon. Figure 1 illustrates an example of forecast.

Indeed, our first contribution is the introduction of *Barycentric Quantiles Interpolation with Piecewise Linear with Exponential Tails (BQ-PLET)* with a higher number of quantiles picked to maintain accuracy in the tail of the distribution. The neural network thus returns 9 pre-determined quantiles. New quantiles in between those are linearly interpolated. Quantiles in the left and right tail are assumed to be described by an exponential distribution. For new mesh points, barycentric interpolation over the 9 quantiles is applied to generate a new PLET distribution.

Finally, our second contribution is the way we apply overrides and adjust the forecast to make it consistent. Indeed, users of the forecast may have reasons to believe that the forecast for a particular

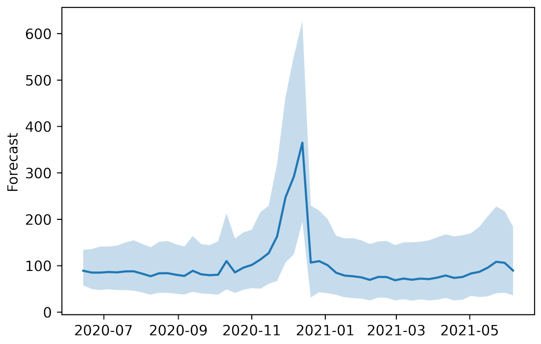

Figure 1: The line corresponds to the median forecast (50th percentile) for the next 52 weeks of demand of a typical product. This is thus representing the forecast at lead times 0 to 51 with span 1 (in weeks) for forecast created on 2020-06-07. The shaded area corresponds to what the forecast expects 90% of the time: the lower bound of that region is the 5th percentile of the forecast, and the upper bound is the 95th percentile.

lead time, span, and quantile is erroneous, e.g. due to a missing likely spike in demand from an upcoming promotion. However, these overrides are a source of inconsistency in the forecast. We thus introduce a linear program that resolves this and makes the forecast consistent.

1 FITTING QUANTILES OVER PERCENTILES

The neural networks described in the previous section fit the inverse cumulative density function but output quantiles that are not guaranteed to be monotonic. However, by modifying the final layer, MQ-CNN and MQ-Transformer can learn the difference between consecutive quantiles, rather than the forecast value. With an appropriate activation function like the Rectified Linear Unit (ReLU), the modified neural network produces quantiles that are guaranteed to be monotonic, as in Park et al. [14]. In related work, Implicit Quantile Networks Wen and Torkkola [20] and Gouttes et al. [8] take an extra input corresponding to the quantile to estimate. In addition, Kan et al. [10] and Huang et al. [9] use the gradient of a convex function to guarantee monotonicity. However, we have observed that sorting is enough, and we focus on this approach here.

PLET. To interpolate and extrapolate new quantiles, Park et al. [15] suggests using piecewise linear interpolation between quantiles fitted empirically, combined with exponential tails for extrapolation. Given monotone quantiles from the neural network, such interpolation and extrapolation is guaranteed to be monotonic. Pickands and Stine [16] also justifies the choice of exponential distribution by showing that this is the maximum-entropy distribution with a given mean and probabilities. We refer to this approach as *Piecewise Linear with Exponential Tails (PLET).*

NRQ. Durkan et al. [4] introduces rational quadratic interpolations that are also monotonic given monotone input to interpolate in between. Linear extrapolation is employed outside of the range of quantiles generated by the neural network, as described in Appendix B. In our case, having an appropriate transformation for the percentiles allows us to also change the behavior of the tails. As such, we use the inverse cumulative distribution function (CDF) of a normal distribution. This transformation preserves monotonicity. We refer to this approach as *Normal Rational Quadratic (NRQ).*

NLP. Fitting a linear polynomial can also preserve monotonicity. Unlike the other algorithms mentioned so far, this is not an interpolation, since the linear function is not guaranteed to pass through the quantiles from the neural network. However, the quantiles are not expected to pass through a straight line without a transformation. We thus again consider mapping the percentiles through the inverse CDF of a normal distribution. We refer to this approach as *Normal Linear Polynomial (NLP)* for quantiles.

NCP and NMCET. A higher degree polynomial fitting of the quantiles is unlikely to be monotonic. Since we expect the quantile distribution to be similar to a normal distribution, we experiment with third degree polynomials on the percentiles mapping through the inverse CDF of a normal distribution. We refer to this approach as *Normal Cubic Polynomial (NCP)* for quantiles. Whilst the result of the NCP algorithm is not guaranteed to be monotonic, there is an algorithm which produces monotonic cubic interpolants Fritsch and Carlson [7] based on Hermite cubic polynomials (quintic in Dougherty et al. [3]), see also Appendix A. We refer to this approach as *Normal Monotone Cubic with Exponential Tails (NMCET).*

TSGD. The TSGD is discussed in the introduction. As a 2-parameter family, it can be fitted through any two quantiles. For comparison with the above, we consider fitting the 50th and 90th percentiles, and the 50th and 99.5th. We refer to these approaches as *TSGD(50,90)* and *TSGD(50,99.5)* respectively.

Figure 2 illustrates the interpolation techniques discussed above.

2 FITTING QUANTILES OVER PERCENTILES AND MESH

We can now extend the fitting algorithms over percentiles to fitting algorithms over both percentiles and mesh.

BM-TSGD. We introduced previously the concept of a forecast horizon mesh, constituted of fixed lead time and span combinations. The mesh is stored with an attached triangulation (currently picked to be a Delaunay triangulation to avoid thin triangles) in order to perform barycentric interpolation (chosen as it provides an interpolation that is continuous across edges) of the mean and variance of the TSGD to obtain quantiles at points not contained within the mesh. We call this approach *Barycentric Moments interpolation for the TSGD (BM-TSGD)*, see Algorithm 1.

BQ-.* Even though quantiles are not linear quantities like the mean, we can still evaluate how barycentric interpolation of quantiles would compare for the TSGD, where we apply barycentric interpolation directly on the quantiles, thus avoiding the back and forth conversion to TSGD moments which has been a performance

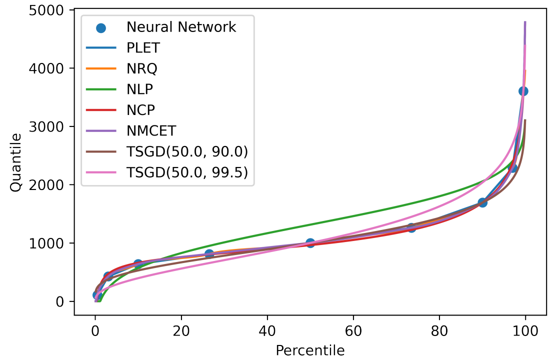

Figure 2: The dots are a representative output of the neural network for a product. Each one of the curves correspond to one of the algorithms described in Section 1. We highlight that PLET, NRQ, and NMCET, interpolate the output of the neural network. TSGD(50.0, 90.0) interpolates the 50th and the 90th percentiles, whereas TSGD(50.0, 99.5) interpolates the 50th and the 99.5th percentiles. NLP and NCP are not guaranteed to pass through any points returned by the neural network. The output of any of the algorithms is clipped to zero if it gets negative.

Algorithm 1 BM-TSGD given a new mesh point P. See Appendix C for more details about computing homogeneous barycentric coordinates.

```
$\mathcal{S} \leftarrow$ simplex in mesh containing new point $P$
$(\lambda_0, \lambda_1, \lambda_2) \leftarrow$ homogeneous barycentric coordinates of $P$ in $\mathcal{S}$ for vertex $i$
for $i = 0, 1, 2$ do
    $(\alpha_i, \theta_i) \leftarrow$ shape and scale for TSGD at vertex $i$
    $(\mu_i, \sigma_i^2) \leftarrow$ mean and variance of TSGD with $(\alpha_i, \theta_i)$
end for
$(\mu_P, \sigma_P^2) \leftarrow \left(\sum_i \lambda_i \mu_i, \sum_i \lambda_i \sigma_i^2\right)$
$(\alpha_P, \theta_P) \leftarrow$ shape and scale of TSGD with $(\mu_P, \sigma_P^2)$
Return desired quantiles from TSGD with shape $\alpha_P$ and scale $\theta_P$
```

bottleneck. We call this approach *Barycentric Quantiles Interpolation of the TSGD.*

In fact, we can extend the idea to more than just the TSGD fitting and combine any of the quantile-fitting with barycentric quantile interpolation: *Barycentric Quantiles interpolations of the PLET/NRQ/NLP/NCP/NMCET* as described in Algorithm 2. We note that quantile-fitting techniques that preserve monotonicity still do so here. In fact, over the quantile-mesh, we are not only interested in monotonicity along the quantile axis, but also the span axis; both of which are preserved here.

M-NLP. Finally, as before, fitting a linear polynomial preserves monotonicity. However, the quantiles are not expected to pass through a quantile-mesh plane without a transformation. We thus again consider mapping the percentiles through the inverse CDF of a normal distribution. We refer to this approach as *Normal Linear Polynomial for the quantile-Mesh.*

Algorithm 2 BQ-PLET given a new mesh point P. Barycentric interpolation can similarly be used with other interpolation in place of PLET, resulting in the family BQ-* of algorithms.

```
$\mathcal{S} \leftarrow$ simplex in mesh containing new point $P$
$(\lambda_0, \lambda_1, \lambda_2) \leftarrow$ homogeneous barycentric coordinates of $P$ in $\mathcal{S}$ for vertex $i$
for $i = 0, 1, 2$ do
    $(q_i^{(1)}, \ldots) \leftarrow$ quantiles for PLET at vertex $i$
end for
$(q_P^{(1)}, \ldots) \leftarrow \left(\sum_i \lambda_i q_i^{(1)}, \ldots\right)$
Return desired quantiles from PLET with quantiles $(q_P^{(1)}, \ldots)$
```

M-NCP. Again, a higher degree polynomial fitting of the quantiles would be unlikely to be monotonic, but not far from monotonic if the higher degrees have small coefficients. As before, we thus consider applying a transformation through the inverse CDF of the normal distribution. We refer to this approach as *Normal Cubic Polynomial for the quantile-Mesh.*

3 EVALUATING FIT WITH REWEIGHTED QUANTILE LOSS

The forecasts are generated via a multiple-stage process wherein two quantiles are predicted by a deep-learning model, which are then used to fit the TSGD. This allows downstream users to obtain a forecast for a given product at any quantile of interest. Yet, not all quantiles are of equal interest – those that sell more units per year. For that reason, we explore a weighting scheme that would enable the optimization of quantiles according to their selling velocity. Two components that would facilitate such a weighting are expected planning horizon (EPH) and critical ratio (CR).

EPH is the time interval between the current buy date plus current vendor lead time (VLT) and the next buy date plus next VLT, measured in days. Intuitively, it is the time span for which we will need to buy enough for to have enough inventory. Buy date is the date on which the purchase order is made and marks the start of the planning period. Vendor lead time is the expected number of days between inventory purchase and arrival.

CR is the percentile at which buying takes place: it is the solution to the News-Vendor Problem that gives the optimal inventory level that we should have. That is, if an in-stock manager opts to buy at a CR of 0.9 over the EPH, the probability of demand being less than or equal to the value of the forecast is 0.9.

Fixing lead time, the density of the joint distribution could be used as the aforementioned weighting scheme for the percentile (CR) and span (analogous to EPH). We now use this density $w_{s,q}$ to define the reweighted quantile loss.

For a forecast $f_{a,l,s,q}$ an products a at quantile q during horizon defined by a lead time l and span s compared against demand $d_{a,l,s}$, the quantile loss is defined as

$$\begin{aligned}&\mathrm{QL}(a,l,s,q)\\ &\qquad := q\cdot(d_{a,l,s}-f_{a,l,s,q})^{+}+(1-q)\cdot(d_{a,l,s}-f_{a,l,s,q})^{-}\end{aligned}$$

with superscripts + and − denote the positive and negative parts, and the demand-weighted quantile loss is

$$\mathrm{WQL}(f,d,l,s,q) := \frac{\sum_a \mathrm{QL}(a,l,s,q)}{\sum_a d_{a,l,s}}$$

as usual. We note that the quantile loss for the 50th quantile (with $q = 0.5$) is equal to the mean absolute error up to a multiplicative constant, and provides an estimate of the median as desired from the application of interest. Using the mean square error would instead lead to an estimate of the mean.

We now reweight this demand-weighted quantile loss using the density $w_{s,q}$. Indeed, we estimate the importance of the span by the frequency of EPH, and that of a quantile, by the frequency of CR. We can thus define the (total) reweighted quantile loss as

$$\mathrm{RQL}(f,d) := \sum_{a,l\in\{0,1\},s,q} w_{s,q}\,\mathrm{WQL}(f,d,l,s,q)$$

using the density $w_{s,q}$. We picked the lead time to be either 0 or 1 in the last equation since most purchase orders are made in the low lead times.

4 EXPERIMENTAL SETUP AND RESULTS

We evaluate our architecture on a demand forecasting problem for a large e-commerce retailer with the objective of producing multi-horizon forecasts for the next 52 weeks. We conduct our experiments on a dataset of millions of products in the US retail store. The models are trained on three years of demand data (2018-2020), and one year (2020-2021) is held out for backtesting. The features used in each are similar to Eisenach et al. [5], Wen et al. [21]. They include static (e.g. product catalog fields), historical (e.g. past demand), and future (e.g. promotions). The model used for control is MQ-Transformer trained only for the 2 percentiles 0.5 and 0.9 and interpolated using BM-TSGD. The architecture proposed is an MQ-Transformer model trained to optimize for 9 percentiles (0.005, 0.03, 0.1, 0.265, 0.5, 0.735, 0.9, 0.97, 0.995). The interpolation strategy we use is the Barycentric Piecewise Linear with Exponential Tails (BQ-PLET).

We evaluate the algorithms on 19 percentiles; the 9 previous ones along with an additional 10: (0.002, 0.01, 0.05, 0.2, 0.35, 0.65, 0.8, 0.95, 0.99, 0.998). We show the weighted quantile loss per percentile in Table 1. The improvements occur mostly for percentiles below the 3rd and above the 97th.

To select BQ-PLET as the interpolation strategy, we looked at the overall reweighted quantile loss and ranked the algorithms in Table 2. Since our downstream customers are most impacted by products that sell most frequently, we assembled this table from a sample of ten thousand products. The difference in the ranking between many of the algorithms is, as expected, small since most agree at the quantiles learned by the neural network. We see that the Barycentric Piecewise Linear with Exponential Tails, also preferred for its simplicity, is among the algorithms with the lowest loss.

Table 1: Difference in weighted quantile loss for BM-TSGD and BQ-PLET over the dataset of millions of products.

Percentile	Difference
0.2	-62.1%
0.5	-43.8%
1	-33.1%
3	-13.4%
5	-9.7%
10	-2.5%
20	-1.0%
26.5	0.4%
35	0.1%
50	**-0.9%**
65	-0.6%
73.5	-2.0%
80	-0.5%
90	**-1.7%**
95	-0.3%
97	-1.9%
99	-4.7%
99.5	-12.6%
99.8	-23.9%

Table 2: Quantile fitting algorithms sorted by ascending reweighted quantile loss over the dataset of ten thousand products.

Algorithm	RQL
BQ-NMCET	-8.2%
BQ-PLET	**-8.2%**
BQ-NCP	-8.2%
BQ-NPLET	-8.1%
BQ-NRQ	-7.5%
BM-TSGD(50.0, 99.5)	-6.4%
BM-TSGD(50.0, 90.0)	**0.0%**
BQ-TSGD(50.0, 90.0)	0.0%
BQ-NLP	1.4%
M-NCP	12.5%
M-NLP	68.8%

5 HANDLING OVERRIDES

Users of the forecast may have reasons to believe that a forecast is erroneous or doesn't know about an upcoming event that would have an impact. For example, they may believe a product forecast doesn't account for a likely spike in demand from an upcoming promotion. Since overrides are common practice, we have been providing a method to the users for adjusting the span-1 forecast on the mesh. Our new methodology must therefore also support this.

As mentioned before, we generate forecast at fixed predetermined combinations of lead time and spans, called the mesh. Users usually expects the forecast to be consistent on the mesh: e.g. the forecast should increase as the span increases, and that horizons

Table 3: Weighted quantile loss for Naive Override as baseline and Consistent Override.

Percentile	Difference
0.2	-95%
0.5	-95%
1	-100%
3	-96%
5	-96%
10	-95%
20	-92%
26.5	-91%
35	-88%
50	-84%
65	-80%
73.5	-75%
80	-72%
90	-57%
95	-45%
97	-24%
99	-2%
99.5	55%
99.8	141%

fully contained in others should have a smaller values. However, that is not guaranteed by the neural networks, and so small such inconsistencies do occur, which are then aggravated by overrides. Finding consistent forecasts on a mesh has garnered recent attention, [12, 13, 19].

Override specification is currently constrained to P50 and P90 quantile estimates, and overriding these values will cause inconsistency across the remaining quantiles of the interpolated cumulative density function. To alleviate this, we replace each lower quantile estimate between P50 and P90 to be the P50 estimate, and lower quantile estimates between P90 and P99 to be the P90 estimate.

We then generate correlated sample paths reflective of user override values using the method detailed in Appendix D. Finally, we replace the forecast distributions with the distribution implied by the overridden quantile values, and run a linear program to generate consistent estimates. Methodological details can be found in Appendix D.

We assess validity of the above methodology relative to the current override strategy – generating the system forecast and directly inputting user override values – on the dataset defined in Section 4. For each product and FCD (the dataset has 52 FCDs), we override the P50 and P90 forecasts to be 10 times their system forecast 10 weeks from their FCD. In assessing the loss, we also assume that the demand value is indeed 10 times larger than it is true value (hence the override was accurate). Estimates from both the current override strategy and our proposed methodology are calculated using the BQ-PLET interpolation described in Section 1. Table 3 summarizes the results in demand weighted quantile loss across all product/FCTs and mesh points when using the override strategy above relative to the naive strategy of only replacing the span-1 forecast.

6 CONCLUSION

We introduced a new distribution-free end-to-end multi-horizon forecasting and vending system; integrating product-level forecasts at fixed lead times, spans and quantiles with piecewise linear interpolation and exponential extrapolation for any other lead time, span or quantile.

The current process keeps the interpolation step as a postprocessing. On one hand, this has the advantage of limiting the per product storage requirement to the output of the neural network. On the other, it does not allow us to learn directly the full quantile distribution inside the neural network; such as Incremental Spline Quantile Function from Park et al. [14] by integrating splines, Implicit Quantile Networks Wen and Torkkola [20] and Gouttes et al. [8] and Temporal Fusion Transformers Lim et al. [11] by adding a new input parameter representing the quantile, and Kan et al. [10] and Huang et al. [9] that also enforce monotonicity. We also can consider other ways to improve forecast with overrides, such as using Neural Copula from Zeng and Wang [22].

REFERENCES

[1] George Athanasopoulos, Rob J. Hyndman, Haiyan Song, and Doris C. Wu. The tourism forecasting competition. *International Journal of Forecasting*, 27(3):822–844, 2011. ISSN 0169-2070. Special Section 1: Forecasting with Artificial Neural Networks and Computational Intelligence Special Section 2: Tourism Forecasting.

[2] Filippo Maria Bianchi, Enrico Maiorino, Michael C Kampffmeyer, Antonello Rizzi, and Robert Jenssen. An overview and comparative analysis of recurrent neural networks for short term load forecasting. *arXiv preprint arXiv:1705.04378*, 2017.

[3] Randall L. Dougherty, Alan Edelman, and James Mac Hyman. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic hermite interpolation. *Mathematics of Computation*, 52:471–494, 1989.

[4] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows, 2019. URL https://arxiv.org/abs/1906.04032.

[5] Carson Eisenach, Yagna Patel, and Dhruv Madeka. MQTransformer: Multi-horizon forecasts with context dependent and feedback-aware attention, 2020.

[6] Corporación Favorita. Corporación favorita grocery sales forecasting. https://www.kaggle.com/c/favorita-grocery-sales-forecasting, 2018.

[7] F. N. Fritsch and R. E. Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246, 1980. doi: 10.1137/0717021. URL https://doi.org/10.1137/0717021.

[8] Adèle Gouttes, Kashif Rasul, Mateusz Koren, Johannes Stephan, and Tofigh Naghibi. Probabilistic time series forecasting with implicit quantile networks, 2021.

[9] Chin-Wei Huang, Ricky T. Q. Chen, Christos Tsirigotis, and Aaron Courville. Convex potential flows: Universal probability distributions with optimal transport and convex optimization, 2021.

[10] Kelvin Kan, François-Xavier Aubet, Tim Januschowski, Youngsuk Park, Konstantinos Benidis, Lars Ruthotto, and Jan Gasthaus. Multivariate quantile function forecaster, 2022.

[11] Bryan Lim, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting, 2019. URL https://arxiv.org/abs/1912.09363.

[12] Dhruv Madeka, Lucas Swiniarski, Dean Foster, Leo Razoumov, Kari Torkkola, and Ruofeng Wen. Sample path generation for probabilistic demand forecasting. In *KDD 2018 Workshop on Mining and Learning from Time Series*, 2018. URL https://www.amazon.science/publications/sample-path-generation-for-probabilistic-demand-forecasting.

[13] Kin G. Olivares, O. Nganba Meetei, Ruijun Ma, Rohan Reddy, Mengfei Cao, and Lee Dicker. Probabilistic hierarchical forecasting with deep poisson mixtures. 2021. doi: 10.48550/ARXIV.2110.13179. URL https://arxiv.org/abs/2110.13179.

[14] Youngsuk Park, Danielle Maddix, François-Xavier Aubet, Kelvin Kan, Jan Gasthaus, and Yuyang Wang. Learning quantile functions without quantile crossing for distribution-free time series forecasting, 2021. URL https://arxiv.org/abs/2111.06581.

[15] Youngsuk Park, Danielle Maddix, François-Xavier Aubet, Kelvin Kan, Jan Gasthaus, and Yuyang Wang. Learning quantile functions without quantile crossing for distribution-free time series forecasting, 2021. URL https://arxiv.org/abs/2111.06581.

[16] James Pickands and Robert A. Stine. Estimation for an m/g/infinite queue with incomplete information. *Biometrika*, 84(2):295–308, 1997. ISSN 00063444. URL http://www.jstor.org/stable/2337458.

[17] Abdulwahed Salam and Abdelaaziz El Hibaoui. Comparison of machine learning algorithms for the power consumption prediction:-case study of tetouan city–. In *2018 6th International Renewable and Sustainable Energy Conference (IRSEC)*, pages 1–5. IEEE, 2018.
[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
[19] Ruofeng Wen and Kari Torkkola. Deep generative quantile-copula models for probabilistic forecasting, 2019. URL https://arxiv.org/abs/1907.10697.
[20] Ruofeng Wen and Kari Torkkola. Deep generative quantile-copula models for probabilistic forecasting, 2019.
[21] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster, 2018.
[22] Zhi Zeng and Ting Wang. Neural copula: A unified framework for estimating generic high-dimensional copula functions, 2022. URL https://arxiv.org/abs/2205.15031.

A APPENDIX: MONOTONE CUBIC INTERPOLATION

Given N knots $\{(x^{(k)}, y^{(k)}\}_1^N$ and $x^{(k)} \leq x \leq x^{(k+1)}$, we let

$$\xi := \frac{x - x^{(k)}}{x^{(k+1)} - x^{(k)}}$$

so that the interpolated quantile $\hat{y}$ at x is

$$\begin{aligned}\hat{y} = y^{(k)} h_{00}(\xi) + y^{(k+1)} h_{01}(\xi) \\ + m_k(x^{(k+1)} - x^{(k)}) h_{10}(\xi) \\ + m_{k+1}(x^{(k+1)} - x^{(k)}) h_{11}(\xi)\end{aligned}$$

where

$$\begin{aligned}h_{00}(t) &= (1+2t)(1-t)^2 \\ h_{10}(t) &= t(1-t)^2 \\ h_{01}(t) &= t^2(3-2t) \\ h_{11}(t) &= t^2(t-1)\end{aligned}$$

are the four Hermite cubic basis.

B APPENDIX: RATIONAL-QUADRATIC INTERPOLATION

We follow Durkan et al. [4]. Given N knots $\{(x^{(k)}, y^{(k)}\}_1^N$ and $x^{(k)} \leq x \leq x^{(k+1)}$, we let

$$\begin{aligned}\xi &:= \frac{x - x^{(k)}}{x^{(k+1)} - x^{(k)}} \\ s^{(k)} &:= \frac{y^{(k+1)} - y^{(k)}}{x^{(k+1)} - x^{(k)}}\end{aligned}$$

so that the interpolated quantile $\hat{y}$ at x is

$$\hat{y} = y^{(k)} + \frac{\left(y^{(k+1)} - y^{(k)}\right)\left[s^{(k)}\xi^2 + \delta^{(k)}\xi(1-\xi)\right]}{s^k + \left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right]\xi(1-\xi)}$$

where $\delta^{(k)} > 0$ is a hyperparameter with $\delta^{(0)} = \delta^{(K)} = 1$.

C APPENDIX: BARYCENTRIC COORDINATES

Given a triangle with vertices $P_0 = (x_0, y_0)$, $P_1 = (x_1, y_1)$, $P_2 = (x_3, y_3)$, and a new point $P = (x, y)$, we are looking for the unique homogeneous barycentric coordinates $(\lambda_0, \lambda_1, \lambda_2)$ satisfying

$$\lambda_0 P_0 + \lambda_1 P_1 + \lambda_2 P_2 = P$$

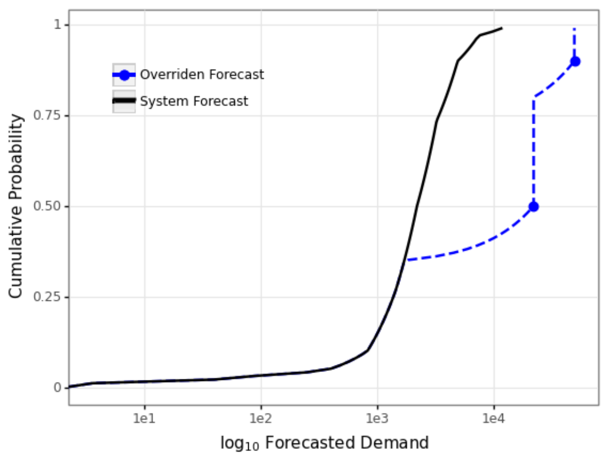

Figure 3: Cumulative probability functions using interpolation for the forecast and an overridden forecast. Forecasted demand is displayed on a log scale.

with $\lambda_0 + \lambda_1 + \lambda_2 = 1$. More explicitly,

$$\begin{aligned}(1 - \lambda_1 - \lambda_2)x_0 + \lambda_1 x_1 + \lambda_2 x_2 = x, \\ (1 - \lambda_1 - \lambda_2)y_0 + \lambda_1 y_1 + \lambda_2 y_2 = y.\end{aligned}$$

or

$$\begin{pmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{pmatrix} \begin{pmatrix} \lambda_0 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix}$$

Since the edges $P_1 - P_0$ and $P_2 - P_0$ of the triangles are linearly independent, the matrix is invertible. Thus, we get

$$\begin{aligned}\begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} &= \begin{pmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{pmatrix}^{-1} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} \\ &= \frac{1}{d} \begin{pmatrix} y_2 - y_0 & x_0 - x_2 \\ x_1 - x_0 & y_0 - y_1 \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix}\end{aligned}$$

where $d = (x_1 - x_0)(y_2 - y_0) - (x_2 - x_0)(y_1 - y_0)$ is the determinant of the matrix, and $\lambda_0 = 1 - \lambda_1 - \lambda_2$.

D APPENDIX: HANDLING OVERRIDES

Figure 3 illustrates this decision, showing the cumulative density function for a single product estimated by interpolated forecast and the forecast with overridden P50 and P90 estimates to be 10 times its forecast.

The result of the override-adjusted algorithm is illustrated in Figure 4. The figure displays historic demand for a single product, where the shaded region describes a 95% prediction interval generated by the forecast after an override is incorporated. Finally, the figure displays the upper envelope – defined by the maximal value at each time point – of 1,000 correlated sample paths generated by the Markovian sample path method and the override-adjusted algorithm. The override-adjusted methodology both incorporates the user override and accurately portrays the forecast distribution for non-overridden values when generating sample paths, alleviating this issue.

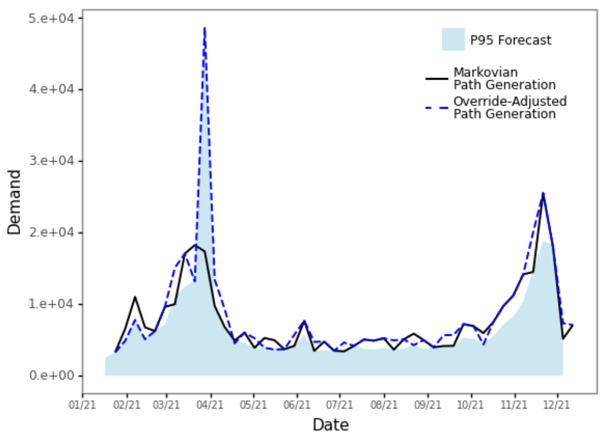

Figure 4: Historic demand for a single product, the override-adjusted 95% prediction interval generated by the forecast, and upper envelopes of 1,000 correlated sample paths generated by the Markovian method and the override-adjusted algorithm.

D.1 Problem Statement

For a demand vector $\underline{X}$ corresponding to the entire forecasted mesh lead-time/span combinations, the task of generating consistent forecasts across the lead-time/span mesh can be formulated as the following constrained optimization problem:

> Let $\underline{X} \in \mathbb{R}^d$ denote a d-dimensional random vector, and let $g_j(\underline{X})$, $j \in [J]$ denote J scalar functions of $\underline{X}$. Given a set of quantiles $Q_{j,k}$ and corresponding probabilities $p_{j,k}$, $k \in [K_j]$ for functions $g_j(\underline{X})$, find the closest consistent quantile distribution for $\underline{X}$.

We define a procedure for vending consistency-corrected demand forecasts using the following steps:

(1) Generate quantile estimates $\widehat{Q}$ from a function $\underline{g}_{[J]}$, whose elements generally enumerate various lead-time span combinations, at percentile values (p_{jk}).
(2) Estimate the cumulative distribution function $\widehat{F}_j(g)$ from $(\widehat{Q}_{j,k})_{k\in[K_j]}$.
(3) Generate L correlated sample paths from $\widehat{F}_j(g)$, $j \in [J]$.
(4) Create an empirical demand distribution $\widehat{F}_{\underline{X}}(x)$ for the latent vector $\underline{X}$ from the L correlated sample paths.
(5) Optimize the objective function $\min \sum_j d(\widehat{F}_j(g), \widehat{F}_{\underline{X}}(\underline{x}))$.

This methodological pipeline requires generation of appropriate correlated sample paths and minimization of the distances $d(\widehat{F}_j(g), \widehat{F}_{\underline{X}}(\underline{x}))$. We describe methods to accomplish these tasks briefly below.

D.2 Generating Correlated Sample Paths

Consistency correction across the lead-time/span mesh requires an underlying distribution assumption on the weekly demand vector $\underline{X}$. However, state-of-the-art models produce only a small collection of quantile knots independently for each element X_w rather than their entire cumulative distribution $F_{X_w}(x)$. Interpolation $\widehat{F}_{X_w}(x)$ of the cumulative distribution function provides estimated marginal distributions for the elements of $\underline{X}$ allowing for direct sampling of the values, but elementwise sampling of the estimated distributions will lead to uncorrelated and hence unrealistic demand vectors.

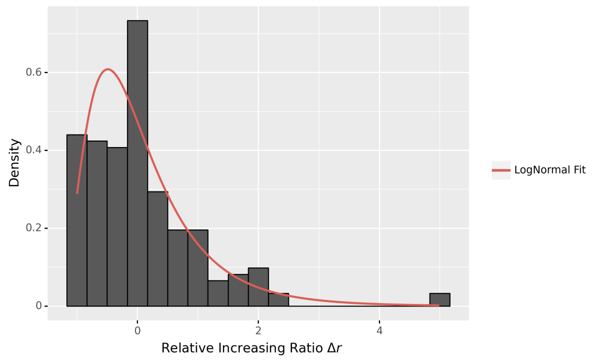

Figure 5: Histogram of historic values r_w from 12/18 - 1/21 fitted to a log normal distribution.

We instead generate correlated sample paths from the interpolated marginal distributions using a Markovian sampling method. Let $\underline{x}^{(H)}$ denote a vector of historic demand and $w = 0$ denote the current week. The *relative increasing rate* of the historic demand vector $\underline{x}^{(H)}$ is defined as

$$r_w := \frac{x^{(H)}_{-w+1} - x^{(H)}_{-w}}{x^{(H)}_{-w}} := \frac{\Delta x^{(H)}_{-w+1}}{x^{(H)}_{-w}}, \tag{1}$$

with corresponding empirical distribution function $\widehat{F}_R(r)$. We then generate L correlated sample paths $\underline{x}^*_l$ by sequentially sampling from the marginal distributions of X_w and accepting according to $\widehat{F}_R(r)$. We detail the method in Algorithm 3.

Algorithm 3 Correlated Sample Path Generation

```
Set $x^*_{-1} \leftarrow x_{-1}$
**for** $w = 0, 1, 2, \ldots$ **do**
    Sample $\widehat{x}_w \sim \widehat{F}_{X_w}(x)$
    Set $\widehat{r}_w \leftarrow \Delta\widehat{x}_w / x^*_{w-1}$
    Sample $u \sim U(0,1)$
    **while** $u > \widehat{F}_R(\widehat{r}_w)$ **do**
        Sample $\widehat{x}_w$ from $\widehat{F}_{X_w}(x)$
        Set $\widehat{r}_w \leftarrow \Delta\widehat{x}_w / x^*_{w-1}$
        Sample $u \sim U(0,1)$
    **end while**
    Set $x^*_w \leftarrow \hat{x}_w$
**end for**
```

Figure 5 depicts the historic relative increasing rates $\hat{r}_w$ for a given product, ranging from −1 to 5 corresponding to a demand of zero (100× demand decrease) and a 5× demand increase respectfully, where the plurality of ratios fall between −1 and 1. While the above formulation generates correlated sample paths, $\widehat{F}_R(r)$ has finite support, so that increases larger than those previously seen will never be accepted. To fix this, we fit a shifted log-normal distribution to the empirical values $\widehat{r}_w$. This produces correlated sample paths with support on $[-1, \infty)$.

When handling forecast overrides, it is important to provide the customer with consistent forecasts that align with their specification. However, empirical distributions generated by the sample path method described above rely on historic demand which may not align with the intended override value. To incorporate the user override, we assume that the user specified value defines a new ground-truth for the cumulative distribution function at the overridden lead-time/span mesh point.

D.3 Linear Program

Estimating consistent quantiles $\widehat{Q}_{jk}$ which minimize distance to the predicted marginal quantiles Q_{jk} can be formulated as a mixed-integer program with complexity $O(JKL)$. Instead, we can solve the reverse-regression problem of finding a probability distribution $\underline{\pi}$ for demand vector $\underline{X}$ which is most likely to generate the predicted marginal quantiles Q_{jk}. Let

$$\mathcal{S}_{jk} = \{l \in [L] \mid \underline{g}_j(\underline{X}_l) \leq Q_{jk}\}, \tag{2}$$

where $\{\underline{x}_l^*\}_{l \in [L]}$ are correlated sample paths generated using Algorithm 3. Let $\underline{\pi} = \{\pi_l\}_{l \in [L]}$ represent the empirical distribution of $\underline{X}$ on the support $\{\underline{x}_l^*\}_{l \in [L]}$. Using the equality

$$\sum_{l \in \mathcal{S}_{jk}} \pi_l = \mathbb{P}(\underline{g}_j(\underline{X}) \leq Q_{jk}),$$

we optimize with respect to $\underline{\pi}$ using the linear program

$$\underline{\pi}^* = \min_{\underline{\pi}} \sum_{j=1}^{J} \sum_{k=1}^{K_j} x_{jk} \tag{3a}$$

such that

$$x_{jk} \geq \sum_{l \in \mathcal{S}_{jk}} \pi_l - p_{jk} \tag{3b}$$

$$x_{jk} \geq p_{jk} - \sum_{l \in S_{jk}} \pi_{jk} \tag{3c}$$

$$\sum_{l=1}^{L} \pi_l = 1, \quad \pi_l \geq 0. \tag{3d}$$

The discrete probability distribution $\underline{\pi}^* \in \mathbb{R}^L$ on each of the correlated sample paths $\underline{x}_l^*$ from minimizing the constrained objective in (3) can then be used to generate consistent quantiles Q_{jk}. To ensure the function $\underline{g}_j(\underline{X}_l)$ agrees with the overridden value, we modify the linear program by adding the constraint

$$\sum_{l \in \mathcal{S}_{jk}^o} \pi_l = \mathbb{P}(\underline{g}_j(\underline{X}) \leq Q_{jk}^o) = p_{jk}, \tag{4}$$

where $\mathcal{S}_{jk}^o = \{l \in [L] \mid \underline{g}_j(\underline{X}_l) \leq Q_{jk}^o\}$.