Bayesian Intermittent Demand Forecasting for Large Inventories

Matthias Seeger, David Salinas, Valentin Flunkert
Amazon Development Center Germany
Krausenstrasse 38
10115 Berlin
matthis@amazon.de, dsalina@amazon.de, flunkert@amazon.de

Abstract

We present a scalable and robust Bayesian method for demand forecasting in the context of a large e-commerce platform, paying special attention to intermittent and bursty target statistics. Inference is approximated by the Newton-Raphson algorithm, reduced to linear-time Kalman smoothing, which allows us to operate on several orders of magnitude larger problems than previous related work. In a study on large real-world sales datasets, our method outperforms competing approaches on fast and medium moving items.

1 Introduction

Demand forecasting plays a central role in supply chain management, driving automated ordering, in-stock management, and facilities planning. Classical forecasting methods, such as exponential smoothing [10] or ARIMA models [5], produce Gaussian predictive distributions. While sufficient for inventories of several thousand fast-selling items, Gaussian assumptions are grossly violated for the extremely large catalogues maintained by e-commerce platforms. There, demand is highly *intermittent* and *bursty*: long runs of zeros, with islands of high counts. Decision making requires quantiles of predictive distributions [14], whose accuracy suffer under erroneous assumptions.

In this work, we detail a novel methodology for intermittent demand forecasting which operates in the industrial environment of a very large e-commerce platform. Implemented in `Apache Spark` [16], our method is used to process many hundreds of thousands of items and several hundreds of millions of item-days. Key requirements are *automated parameter learning* (no expert interventions), *scalability* and a high degree of operational *robustness*. Our system produces forecasts both for short (one to three weeks) and longer lead times (up to several months), the latter require feature maps depending on holidays, sales days, promotions, and price changes. Previous work on intermittent demand forecasting in Statistics is surveyed in [15]: none of these address longer lead times. On a modelling level, our proposal is related to [6], yet several novelties are essential for operating at the industrial scale we target here. This paper makes the following contributions:

- A combination of generalized linear models and time series smoothing. The former enables medium and longer term forecasts, the latter provides temporal continuity and reasonable distributions over time. Compared to [6], we provide empirical evidence for the usefulness of this combination.
- A novel algorithm for maximum likelihood parameter learning in state space models with non-Gaussian likelihood, using approximate Bayesian inference. While there is substantial related prior work, our proposal stands out in robustness and scalability. We show how approximate inference is solved by the Newton-Raphson algorithm, fully reduced to Kalman smoothing once per iteration. This reduction scales *linearly* (a vanilla implementation

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

would scale cubically). While previously used in Statistics [7, Sect. 10.7], this reduction is not widely known in Machine Learning. If L-BFGS is used instead (as proposed in [6]), approximate inference fails in our real-world use cases.

- A multi-stage likelihood, taylored to intermittent and bursty demand data (extension of [15]), and a novel transfer function for Poisson likelihood, which robustifies the Laplace approximation for bursty data. We demonstrate that our approach would not work without these novelties.

The structure of this paper is as follows. In Section 2, we introduce intermittent demand likelihood function as well as a generalized linear model baseline. Our novel *latent state forecasting* methodology is detailed in Section 3. We relate our approach to prior work in Section 4. In Section 5, we evaluate our methods both on publicly available data and on a large dataset of real-world demand in the context of e-commerce, comparing against state of the art intermittent forecasting methods.

2 Generalized Linear Models

In this section, we introduce a likelihood function for intermittent demand data, along with a generalized linear model as baseline. Denote demand by $z_{it} \in \mathbb{N}$ (i for item, t for day). Our goal is to predict *distributions* of z_{it} aggregates in the future. We do this by fitting a probabilistic model to maximize the likelihood of training demand data, then drawing sample paths from the fitted model, which represent forecast distributions. In the sequel, we fix an item i and write z_t instead of z_{it}.

A model is defined by a *likelihood* $P(z_t|y_t)$ and a latent function y_t. An example is the *Poisson*:

$$P_{\text{poi}}(z|y) = \frac{1}{z!}\lambda(y)^z e^{-\lambda(y)}, \quad z \in \mathbb{N}, \tag{1}$$

where the rate $\lambda(y)$ depends on y through a transfer function. Demand data over large inventories is both intermittent (many $z_t = 0$) and bursty (occasional large z_t), and is not well represented by a Poisson. A better choice is the *multi-stage likelihood*, generalizing a proposal in [15]. This likelihood decomposes into $K = 3$ stages, each with its latent function $y^{(k)}$. In stage $k = 0$, we emit $z = 0$ with probability[1] $\sigma(y^{(0)})$. Otherwise, we transfer to stage $k = 1$, where $z = 1$ is emitted with probability $\sigma(y^{(1)})$. Finally, if $z \geq 2$, then stage $k = 2$ draws $z - 2$ from the Poisson (1) with rate $\lambda(y^{(2)})$.

If the latent function y_t (or functions ${y_t}^{(k)}$) is linear, $y_t = \boldsymbol{x}_t^\top \boldsymbol{w}$, we have a *generalized linear model* (GLM) [11]. Features in $\boldsymbol{x}_t$ include kernels anchored at holidays (Christmas, Halloween), seasonality indicators (DayOfWeek, MonthOfYear), promotion or price change indicators. The weights $\boldsymbol{w}$ are learned by maximizing the training data likelihood. For the multi-stage likelihood, this amounts to separate instances of binary classification at stages 0, 1, and Poisson regression at stage 2. Generalized linear forecasters work reasonably well, but have some important drawbacks. They lack temporal continuity: for short term predictions, even simple smoothers can outperform a tuned GLM. More important, a GLM predicts overly narrow forecast distributions, whose widths do not grow over time, and it neglects temporal correlations. Both drawbacks are alleviated in Gaussian linear time series models, such as *exponential smoothing* (ES) [10]. A major challenge is to combine this technology with general likelihood functions (Poisson, multi-stage) to enable intermittent demand forecasting.

3 Latent State Forecasting

In this section, we develop latent state forecasting for intermittent demand, combining GLMs, general likelihoods, and exponential smoothing time series models. We begin with a single likelihood $P(z_t|y_t)$, for example the Poisson (1), then consider a multi-stage extension. The latent process is

$$y_t = \boldsymbol{a}_t^\top \boldsymbol{l}_{t-1} + b_t, \quad b_t = \boldsymbol{w}^\top \boldsymbol{x}_t, \quad \boldsymbol{l}_t = \boldsymbol{F}\boldsymbol{l}_{t-1} + \boldsymbol{g}_t\varepsilon_t, \quad \varepsilon_t \sim N(0,1). \tag{2}$$

Here, b_t is the GLM deterministic linear function, $\boldsymbol{l}_t$ is a *latent state*. This *innovation state space model* (ISSM) [10] is defined by $\boldsymbol{a}_t$, $\boldsymbol{g}_t$ and $\boldsymbol{F}$, as well as the prior $\boldsymbol{l}_0 \sim P(\boldsymbol{l}_0)$. Note that ISSMs are characterized by a *single* Gaussian innovation variable ε_t per time step. In our experiments here, we

[1] Here, $\sigma(u) := (1 + e^{-u})^{-1}$ is the logistic sigmoid.

employ a simple[2] instance:

$$y_t = l_{t-1} + b_t, \quad l_t = l_{t-1} + \alpha\varepsilon_t, \quad l_0 \sim N(\mu_0, \sigma_0^2),$$

meaning that $\boldsymbol{F} = [1]$, $\boldsymbol{a}_t = [1]$, $\boldsymbol{g}_t = [\alpha]$, and the latent state contains a level component only. The free parameters are $\boldsymbol{w}$ (weights), $\alpha > 0$, and $\mu_0, \sigma_0 > 0$ of $P(l_0)$, collected in the vector $\boldsymbol{\theta}$.

3.1 Training. Prediction. Multiple Stages

We would like to learn $\boldsymbol{\theta}$ by maximizing the likelihood of data $[z_t]_{t=1,\dots,T}$. Compared to the GLM case, this is harder to do, since latent (unobserved) variables $\boldsymbol{s} = [\varepsilon_1, \dots, \varepsilon_{T-1}, \boldsymbol{l}_0{}^\top]^\top$ have to be integrated out. If our likelihood $P(z_t|y_t)$ was Gaussian, this marginalization could be computed analytically via Kalman smoothing [10]. With a non-Gaussian likelihood, the problem is analytically intractable, yet is amenable to the *Laplace approximation* [4, Sect. 4.4]. The exact log likelihood is $\log P(\boldsymbol{z}|\boldsymbol{\theta}) = \log \int P(\boldsymbol{z}, \boldsymbol{s}|\boldsymbol{\theta})\, d\boldsymbol{s} = \log \int \prod_t P(z_t|y_t) P(\boldsymbol{s})\, d\boldsymbol{s}$, where $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{s})$ is the affine mapping given by (2). We proceed in two steps. First, we find the *mode* of the posterior: $\hat{\boldsymbol{s}} = \operatorname{argmax} \log P(\boldsymbol{z}, \boldsymbol{s}|\boldsymbol{\theta})$, the *inner optimization* problem. Second, we replace $-\log P(\boldsymbol{z}, \boldsymbol{s}|\boldsymbol{\theta})$ by its quadratic Taylor approximation $f(\boldsymbol{s}; \boldsymbol{\theta})$ at the mode. The criterion to replace the negative log likelihood is $\psi(\boldsymbol{\theta}) := -\log \int e^{-f(\boldsymbol{s};\boldsymbol{\theta})}\, d\boldsymbol{s}$. More precisely, denote $\phi_t(y_t) := -\log P(z_t|y_t)$, and let $\hat{\boldsymbol{y}} = \boldsymbol{y}(\hat{\boldsymbol{s}})$, where $\hat{\boldsymbol{s}}$ is the posterior mode. The log-concavity of the likelihood implies that $\phi_t(y_t)$ is convex, and $\phi_t''(y_t) > 0$. The quadratic Taylor approximation to $\phi_t(y_t)$ at $\hat{y}_t$ is $\tilde{\phi}_t(y_t) := -\log N(\tilde{z}_t|y_t, \sigma_t^2)$, where $\sigma_t^2 = 1/\phi_t''(\hat{y}_t)$ and $\tilde{z}_t = \hat{y}_t - \sigma_t^2 \phi_t'(\hat{y}_t)$. Now, Laplace's approximation to $-\log P(\boldsymbol{z}|\boldsymbol{\theta})$ can be written as

$$\psi(\boldsymbol{\theta}) = -\log \int \prod_t N(\tilde{z}_t|y_t, \sigma_t^2) P(\boldsymbol{s})\, d\boldsymbol{s} + \sum_t \left(\phi_t(\hat{y}_t) - \tilde{\phi}_t(\hat{y}_t)\right), \quad \boldsymbol{y} = \boldsymbol{y}(\boldsymbol{s}; \boldsymbol{\theta}). \tag{3}$$

For log-concave[3] $P(z_t|y_t)$, the inner optimization is a convex problem. We use the *Newton-Raphson* algorithm to solve it. This algorithm iterates between fitting the current criterion by its local second order approximation and minimizing the quadratic surrogate. For the former step, we compute y_t values by a forward pass (2), then replace the potentials $P(z_t|y_t)$ by $N(\tilde{z}_t|y_t, \sigma_t^2)$, where the values $\tilde{z}_t$, σ_t^2 are determined by the second order fit (as above, but $\hat{y}_t \to y_t$). The latter step amounts to computing the posterior mean (equal to the mode) $\mathrm{E}[\boldsymbol{s}]$ of the resulting Gaussian-linear model. This inference problem is solved by Kalman smoothing.[4]

Not only finding the mode $\hat{\boldsymbol{s}}$, but also the computation of $\nabla_{\boldsymbol{\theta}} \psi$, is fully reduced to Kalman smoothing. This point is crucial. We can find $\hat{\boldsymbol{s}}$ by the most effective optimization algorithm there is. In general, each Newton step requires the $O(T^3)$ inversion of a Hessian matrix. We reduce it to Kalman smoothing, a robust algorithm with $O(T)$ scaling. As shown in Section 4, Newton-Raphson is essential here: other commonly used optimizers fail to find $\hat{\boldsymbol{s}}$ in a reasonable time.

Prediction samples are obtained as follows. Denote observed demand by $[z_1, z_2, \dots, z_T]$, unobserved demand in the prediction range by $[z_{T+1}, z_{T+2}, \dots]$. We run Newton-Raphson one more time to obtain the Gaussian approximation to the posterior $P(\boldsymbol{l}_T|\boldsymbol{z}_{1:T})$ over the final state. For each sample path $[z_{T+t}]$, we draw $\boldsymbol{l}_T \sim P(\boldsymbol{l}_T|\boldsymbol{z}_{1:T})$, $\varepsilon_{T+t} \sim N(0, 1)$, compute $[y_{T+t}]$ by a forward pass, and $z_{T+t} \sim P(z_{T+t}|y_{T+t})$. Drawing prediction samples is not more expensive than from a GLM.

Finally, we generalize latent state forecasting to the multi-stage likelihood. As for the GLM, we learn parameters $\boldsymbol{\theta}^{(k)}$ separately for each stage k. Stages $k = 0, 1$ are binary classification, while stage $k = 2$ is count regression. Say that a day t is *active* at stage k if $z_t \geq k$. Recall that with GLMs, we simply drop non-active days. Here, we use ISSMs for $[y_t^{(k)}]$ on the full range $t = 1, \dots, T$, but all non-active $y_t^{(k)}$ are considered *unobserved*: no likelihood potential is associated with t. Both Kalman smoothing and mode finding (Laplace approximation) are adapted to missing observations, which presents no difficulties (see also Section 5.1).

[2] More advanced variants include damping, linear trend, and seasonality factors [10].

[3] Unless otherwise said, all likelihoods in this paper are log-concave.

[4] We use a numerically robust implementation of Kalman smoothing, detailed in [10, Sect. 12].

3.2 Some Details

In this section, we sketch technical details, most of which are novel contributions. As demonstrated in our experiments, these details are essential for the whole approach to work robustly at the intended scale on our difficult real-world data. Full details are given in a supplemental report.

We use L-BFGS for the *outer optimization* of $\psi(\boldsymbol{\theta})$, encoding the constrained parameters: $\alpha = \alpha_m + (\alpha_M - \alpha_m)\sigma(\theta_1)$; $0 < \alpha_m < \alpha_M$; $\sigma_0 = \log(1 + e^{\theta_2}) > 0$. We add a quadratic regularizer $\sum_j (\rho_j/2)(\theta_j - \bar{\theta}_j)^2$ to the criterion, where ρ_j, $\bar{\theta}_j$ are shared across all items. Finally, recall that with the multi-stage likelihood, day t is unobserved at stage $k > 1$ if $z_t < k$. If for some item, there are less than 7 observed days in a stage, we skip training and *fall back* to fixed parameters $\bar{\boldsymbol{\theta}}$.

Every single evaluation of $\psi(\boldsymbol{\theta})$ requires finding the posterior mode $\hat{\boldsymbol{s}}$. This high-dimensional inner optimization has to converge robustly in few iterations: $\hat{\boldsymbol{s}} = \operatorname{argmin} F(\boldsymbol{s}; \boldsymbol{\theta}) := -\log P(\boldsymbol{z}|\boldsymbol{s}) - \log P(\boldsymbol{s}) = \sum_t \phi_t(y_t) - \log P(\boldsymbol{s})$. The use of Newton-Raphson and its reduction to linear-time Kalman smoothing was noted above. The algorithm is extended by a line search procedure as well as a heuristic to pick a starting point $\boldsymbol{s}_0$ (see supplemental report).

We have to compute the gradient $\nabla_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})$, where the criterion is given by (3). The main difficulty here are indirect dependencies: $\psi(\boldsymbol{\theta}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{s}})$, where $\hat{\boldsymbol{y}} = \boldsymbol{y}(\hat{\boldsymbol{s}}; \boldsymbol{\theta})$, $\hat{\boldsymbol{s}} = \hat{\boldsymbol{s}}(\boldsymbol{\theta})$. Since $\hat{\boldsymbol{s}}$ is computed by an iterative algorithm, commonly used automated differentiation tools do not sensibly apply here. Maybe the most difficult indirect term is $(\partial_{\hat{\boldsymbol{s}}} \psi)^\top (\partial \hat{\boldsymbol{s}} / \partial \theta_j)$, where $\theta_j \in \boldsymbol{\theta}$. First, $\hat{\boldsymbol{s}}$ is defined by $\partial_{\hat{\boldsymbol{s}}} F = \boldsymbol{0}$. Taking the derivative w.r.t. θ_j on both sides, we obtain $(\partial \hat{\boldsymbol{s}} / \partial \theta_j) = -(\partial_{\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}} F)^{-1} \partial_{\hat{\boldsymbol{s}}, \theta_j} F$, so we are looking at $-(\partial_{\hat{\boldsymbol{s}}, \theta_j} F)^\top (\partial_{\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}} F)^{-1} (\partial_{\hat{\boldsymbol{s}}} \psi)$. It is of course out of the question to compute and invert $\partial_{\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}} F$. But $(\partial_{\hat{\boldsymbol{s}}, \hat{\boldsymbol{s}}} F)^{-1} (\partial_{\hat{\boldsymbol{s}}} \psi)$ corresponds to the *posterior mean* for an ISSM with Gaussian likelihood, which depends on $\partial_{\hat{\boldsymbol{s}}} \psi$. This means that the indirect gradient part costs *one more* run of Kalman smoothing, independent of the number of parameters θ_j. Note that the same reasoning underlies our reduction of Newton-Raphson to Kalman smoothing.

A final novel contribution is essential for making the Laplace approximation work on real-world bursty demand data. Recall the transfer function $\lambda(y)$ for the Poisson likelihood (1) at the highest stage $k = 2$. As shown in Section 4, the exponential choice $\lambda = e^y$ fails for all but short term forecasts. With a GLM, the logistic transfer $\lambda(y) = g(y)$ works well, where $g(u) := \log(1 + e^u)$. It behaves like e^y for $y < 0$, but grows linearly for positive y. However, it exhibits grave problems for latent state forecasting. Denote $\phi(y) := -\log P(z|y)$, where $P(z|y)$ is the Poisson with logistic transfer. Recall Laplace's approximation from Section 3.1: $\phi(\cdot)$ is fit by a quadratic $\tilde{\phi}(\cdot) = (\cdot - \tilde{z})/(2\sigma^2)$, where $\sigma^2 = 1/\phi''(y)$, $\tilde{z} = y - \sigma^2 \phi'(y)$. For large y *and* $z = 0$, these two terms scale as e^y, while for $z > 0$ they grow polynomially. In real-world data, we regularly exhibit sizable counts (say, a few $z_t > 25$, driving up y_t), followed by a single $z_t = 0$. At this point, huge values $(\tilde{z}_t, \sigma_t^2)$ arise, causing cancellation errors in $\psi(\boldsymbol{\theta})$, and the outer optimization terminates prematurely.

The root cause for these issues lies with the transfer function: $g(y) \approx y$ for large y, its curvature behaves as e^{-y}. Our remedy is to propose the novel *twice logistic transfer function*: $\lambda(y) = g(y(1 + \kappa g(y))$, where $\kappa > 0$. If $\phi^\kappa(y) = -\log P(z|y)$ with the new transfer function, then $\phi^\kappa(y)$ behaves similar to $\phi(y)$ for small or negative y, but crucially $(\phi^\kappa)''(y) \approx 2\kappa$ for large y *and any* $z \in \mathbb{N}$. This means that Laplace approximation terms are $O(1/\kappa)$. Setting $\kappa = 0.01$ resolves all problems described above. Importantly, the resulting Poisson likelihood is log-concave for any $\kappa \geq 0$. We conjecture that similar problems may arise with other "local" variational or expectation propagation inference approximations as well. The twice logistic transfer function should therefore be of wider applicability.

4 Related Work

Our work has precursors both in Statistics and Machine Learning. Maximum likelihood learning for exponential smoothing models is developed in [10]. These methods are limited to Gaussian likelihood, approximate Bayesian inference is not used. Starting from Croston's method [10, Sect. 16.2], there is a sizable literature on intermittent demand forecasting, as reviewed in [15]. The best-performing method in [15] uses negative binomial likelihood and a damped dynamic, parameters are learned by maximum likelihood. There is no latent (random) state, and neither non-Gaussian inference nor Kalman smoothing are required. It does not allow for a combination with GLMs.

We employ approximate Bayesian inference in a linear dynamical system, for which there is a lot of prior work in Machine Learning [3, 1, 2]. While Laplace's technique is the most frequently used deterministic approximation in Statistics, both in publications and in automated inference systems [13], other techniques such as expectation propagation are applicable to models of interest here [12, 8]. The robustness and predictable running time of Laplace's approximation are key in our application, where inference is driving parameter learning, running in parallel over hundreds of thousands of items. Expectation propagation is not guaranteed to converge, and Markov chain Monte Carlo methods even lack automated convergence tests.

The work most closely related to ours is [6]. They target intermittent demand forecasting, using a Laplace approximation for maximum likelihood learning, allow for a combination with GLMs, and go beyond our work transferring information between items by way of a hierarchical prior distribution. Their work is evaluated on small datasets and short term scenarios only. In contrast, our system runs robustly on many hundreds of thousands of items and many millions of item-days, a three orders of magnitude larger scale than what they report. They do not explore the value of a feature-based deterministic part, which on our real-world data is essential for medium term forecasts. We find that a number of choices in [6] are limiting when it comes to robustness and scalability. First, they choose a likelihood which is not log-concave for two reasons: they use a negative binomial distribution instead of a Poisson, and they use zero-inflation instead of a multi-stage setup.[5] This means their inner optimization problem is non-convex, jeopardizing robustness and efficiency of the nested learning process. Moreover, in our multi-stage setup, the *conditional probability* of $z_t = 0$ versus $z_t > 0$ is represented exactly, while zero-inflation caters for a time-independent zero probability only.

Next, they use an exponential transfer function $\lambda = e^y$ for the negative binomial rate, while we propose the novel twice logistic function (Section 3.2). Experiments with the exponential choice on our data resulted in total failure, at least beyond short term forecasts. Its huge curvature for large y results in extremely large and instable predictions around holidays. In fact, the exponential function causes rapid growth of predictions even without a linear function extension, unless the random process is strongly damped. Finally, they use a standard L-BFGS solver for their inner problem, evaluating the criterion using additional sparse matrix software. In contrast, we enable Newton-Raphson by reducing it to Kalman smoothing. In Figure 1, we evaluate the usefulness of L-BFGS for mode finding in our setup.[6] L-BFGS clearly fails to attain decent accuracy in any reasonable amount of time, while Newton-Raphson converges reliably. Such inner reliability is key to reaching our goal of fully automated learning in an industrial system. In conclusion, while the lack of public code for [6] precludes a direct comparison, their approach, while partly more advanced, should be limited to smaller problems, shorter forecast horizons, and would be hard to run in an industrial setting.

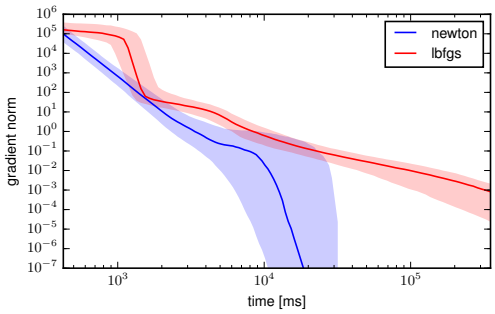

Figure 1: Comparison Newton-Raphson vs. L-BFGS for inner optimization. Sampled at first evaluation of $\psi(\boldsymbol{\theta})$. Shown are median (p10, p90) over ca. 1500 items. L-BFGS fails to converge to decent accuracy.

5 Experiments

In this section, we present experimental results, comparing variants of our approach to related work.

5.1 Out of Stock Treatment

With a large and growing inventory, a fraction of items is *out of stock* at any given time, meaning that order fulfillments are delayed or do not happen at all. When out of stock, an item cannot be sold

[5] Zero-inflation, $p_0 \mathrm{I}_{\{z_t=0\}} + (1-p_0)P'(z_t|y_t)$, destroys log-concavity for $z_t = 0$.

[6] The inner problem is convex, its criterion is efficiently implemented (no dependence on foreign code). The situation in [6] is likely more difficult.

($z_t = 0$), yet may still elicit considerable customer demand. The probabilistic nature of latent state forecasting renders it easy to use out of stock information. If an item is not in stock at day t, the data $z_t = 0$ is explained away, and the corresponding likelihood term should be dropped. As noted in Section 3.1, this presents no difficulty in our framework.

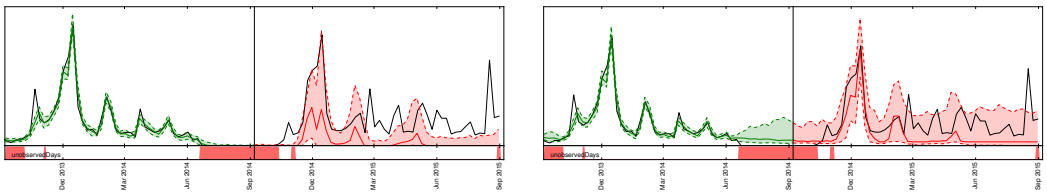

Figure 2: Demand forecast for an item which is partially out of stock. Each panel: Training range left (green), prediction range right (red), true targets black. In color: Median, P10 to P90. Bottom: Out of stock ($\geq 80\%$ of day) marked in red. **Left**: Out of stock signal ignored. Demand forecast drops to zero, strong underbias in prediction range. **Right**: Out of stock regions treated as missing observations. Demand becomes uncertain in out of stock region. No underbias in prediction range.

In Figure 2, we show demand forecasts for an item which is out of stock during certain periods in the training range. It is obvious that ignoring the out of stock signal leads to systematic underbias (since $z_t = 0$ is interpreted as "no demand"). This underbias is corrected for by treating out of stock regions as having unobserved targets. Note that an item may be partially out of stock during a day, still creating some sales. In such cases, we could treat z_t as unobserved, but lower-bounded by the sales, and an expectation maximization extension may be applied. However, such situations are comparatively rare in our data (compared to full-day out of stock). In the rest of this section, latent state forecasting is taking out of stock information into account.

5.2 Comparative Study

We present experimental results obtained on a number of datasets, containing intermittent counts time series. `Parts` contains monthly demand of spare parts at a US automobile company, is publicly available, and was previously used in [10, 15, 6]. Further results are obtained on internal daily e-commerce sales data. In either case, we subsampled the sets in a stratified manner from a larger volume used in our production setting. `EC-sub` is medium size and contains fast and medium moving items. `EC-all` is a large dataset (more than 500K items, 150M item-days), being the union of `EC-sub` with items which are slower moving. Properties of these datasets are given in Figure 3, top left. Demand is highly intermittent and bursty in all cases, as witnessed by a large CV^2 and a high proportion of $z_t = 0$: these properties are typical for supply chain data. Not only is `EC-all` much larger than any public demand forecasting dataset we are aware of, our internal datasets consists of longer series (up to $10\times$) and are more bursty than `Parts`.

The following methods are compared. `ETS` is exponential smoothing with Gaussian additive errors and automatic model selection, a frequently used R package [9]. `NegBin` is our implementation of the negative binomial damped dynamic variant of [15]. We consider two variants of our latent state forecaster: `LS-pure` without features, and `LS-feats` with a feature vector $\boldsymbol{x}_t$ (basic seasonality, kernels at holidays, price changes, out of stock). Predictive distributions are represented by 100 samples over the prediction range (length 8 for `Parts`, length 365 for others). We employ quadratic regularization for all methods except `ETS` (see Section 3.2). Hyperparameters consist of regularization constants ρ_j and centers $\bar{\theta}_j$ (full details are given in the supplemental report). We tune[7] such parameters on random 10% of the data, evaluating test results on the remaining 90%. For `LS-pure` and `LS-feats`, we use two sets of tuned hyperparameters on the largest set `EC-all`: one for the `EC-sub` part, the other for the rest.

Our metrics quantify the forecast accuracy of certain quantiles of predictive distributions. They are defined in terms of *spans* $[L, L + S)$ in the prediction range, where L are *lead times*. In general, we ignore days when items are out of stock (see Figure 3, top left, for in-stock ratios).

[7] We found that careful hyperparameter tuning is important for obtaining good results, also for `NegBin`. In contrast, regularization is not even mentioned in [15] (our implementation of `NegBin` includes the same quadratic regularization as for our methods).

If $\pi_{it} = \mathrm{I}_{\{i \text{ in stock at } t\}}$, define $Z_{i;(L,S)} = \sum_{t=L}^{L+S-1} \pi_{it} z_{it}$. For $\rho \in (0,1)$, the predicted ρ-quantile of $Z_{i;(L,S)}$ is denoted by $\hat{Z}^{\rho}_{i;(L,S)}$. These predictions are obtained from the sample paths by first summing over the span, then estimating the quantile by way of sorting. The *ρ-quantile loss*[8] is defined as $L_\rho(z,\hat{z}) = 2(z-\hat{z})(\rho \mathrm{I}_{\{z>\hat{z}\}} - (1-\rho)\mathrm{I}_{\{z\le\hat{z}\}})$. The *$P(\rho\cdot 100)$ risk metric* for $[L, L+S)$ is defined as $R^{\rho}[\mathcal{I};(L,S)] = |\mathcal{I}|^{-1} \sum_{i\in\mathcal{I}} L_\rho(Z_{i;(L,S)}, \hat{Z}^{\rho}_{i;(L,S)})$, where the left argument $Z_{i;(L,S)}$ is computed from test targets.[9] We focus on P50 risk ($\rho = 0.5$; mean absolute error) and P90 risk ($\rho = 0.9$; the 0.9-quantile is often relevant for automated ordering).

	Parts	EC-sub	EC-all
# items	19874	39700	534884
Unit t	month	day	day
Median CV^2	2.4	5.8	9.7
Freq. $z_t = 0$	54%	46%	83%
In-stock ratio	100%	73%	71%
Avg. size series	33	329	293
# item-days	656K	13M	157M

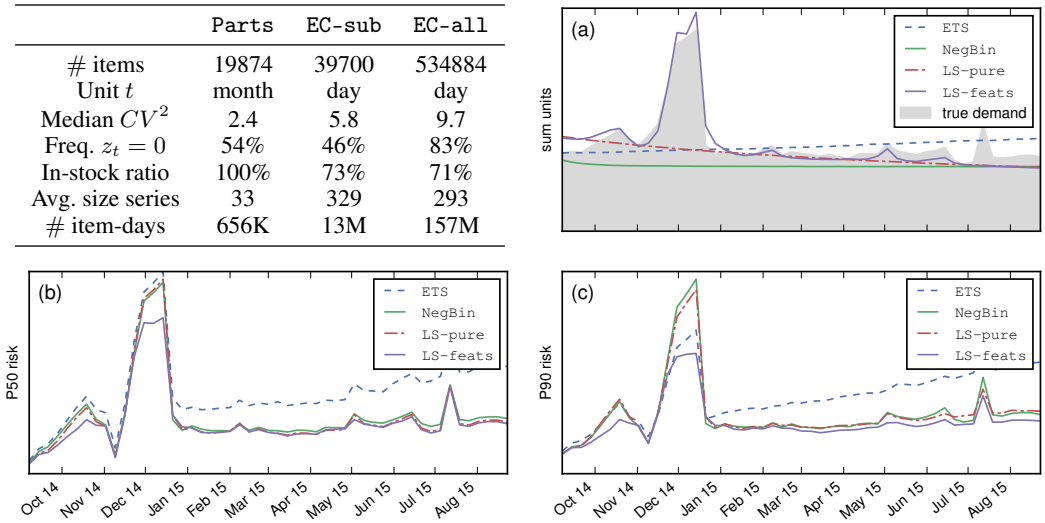

Figure 3: **Table**: Dataset properties. $CV^2 = \mathrm{Var}[z_t]/\mathrm{E}[z_t]^2$ measures burstiness. **(a)**: Sum of weekly P50 point (median) forecast over a one-year prediction range for the different methods (lines) as well as sum of true demand (shaded area), on dataset $\mathcal{I} =$ `EC-sub`. **(b)**: Weekly P50 risk $R^{0.5}[\mathcal{I};(7\cdot k, 7)]$, $k = 0, 1, \ldots$, for same dataset. **(c)**: Same as (b) for P90 risk.

We plot the P50 and P90 risk on dataset `EC-sub`, as well the sum of P50 point (median) forecast and the true demand, in the three panels of Figure 3. All methods work well in the first week, but there are considerable differences further out. Naturally, losses are highest during the Christmas peak sales period. `LS-feats` strongly outperforms all others in this critical region (see Figure 3, top right), by means of its features (holidays, seasonality). The Gaussian predictive distributions of `ETS` exhibit growing errors over time. With the exception of the Christmas period, `NegBin` works rather well (in particular in P50 risk), but is uniformly outperformed by both `LS-pure`, and `LS-feats` in particular.

A larger range of results are given in Table 1 (`Parts`, `EC-sub`) and Table 2 (`EC-all`), where numbers are relative to `NegBin`. Note that the R code for `ETS` could not be run on the large `EC-all`. On `Parts`, `NegBin` works best, yet `LS-pure` comes close (we did not use features on this dataset). On `EC-sub`, `LS-feats` outperforms all others in all scenarios. The featureless `NegBin` and `LS-pure` are comparable on this dataset. On the largest set `EC-all`, `LS-feats` generally outperforms the others, but differences are smaller.

Finally, we report running times of parameter learning (outer optimization) for `LS-feats` on `EC-sub`. L-BFGS was run with `maxIters = 55`, `gradTol = 10^-5`. Our experimental cluster consists of about 150 nodes, with `Intel Xeon E5-2670` CPUs (4 cores) and 30GB RAM. Profiling was done separately in each stage: $k = 0$ ($P5 = 0.180s$, $P50 = 1.30s$, $P95 = 2.15s$), $k = 1$ ($P5 = 0.143s$, $P50 = 1.11s$, $P95 = 1.79s$), $k = 2$ ($P5 = 0.138s$, $P50 = 1.29s$, $P95 = 3.25s$). Here, we quote median (P50), 5% and 95% percentiles (P5, P95). The largest time recorded was $10.4s$. The narrow spread of these numbers witnesses the robustness and predictability of the nested optimization process, crucial properties in the context of production systems running on parallel compute clusters.

[8] $\mathrm{E}_Z[L_\rho(Z,\hat{z})]$ is minimized by the ρ-quantile. Also, $L_{0.5}(z,\hat{z}) = |z-\hat{z}|$.

[9] More precisely, we filter $\mathcal{I}$ before use in $R^{\rho}[\mathcal{I};(L,S)]$: $\mathcal{I}' = \{i \in \mathcal{I} \mid \sum_{t=L}^{L+S-1} \pi_{it} \ge 0.8S\}$.

	Parts				EC-sub					
	P90 risk		P50 risk		P90 risk			P50 risk		
(L, S)	$(0, 2)$	dy(8)	$(0, 2)$	dy(8)	$(0, 56)$	$(21, 84)$	wk(33)	$(0, 56)$	$(21, 84)$	wk(33)
ETS	1.04	1.04	1.19	1.38	0.99	0.75	1.13	1.07	1.10	1.18
LS-pure	1.08	1.06	1.04	1.06	1.07	0.97	0.99	0.95	1.03	0.99
LS-feats	–	–	–	–	**0.80**	**0.73**	**0.85**	**0.84**	**0.84**	**0.94**
NegBin	**1.00**	**1.00**	**1.00**	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00

Table 1: Results for dataset Parts (left) and EC-sub (right). Metric values relative to NegBin (each column). dy(8): Average of $R^\rho[\mathcal{I}; (k, 1)]$, $k = 0, \ldots, 7$. wk(33): Average of $R^\rho[\mathcal{I}; (7 \cdot k, 7)]$, $k = 0, \ldots, 32$.

	P90 risk			P50 risk		
(L, S)	$(0, 56)$	$(21, 84)$	wk(33)	$(0, 56)$	$(21, 84)$	wk(33)
LS-pure	1.11	1.03	0.99	1.00	1.03	1.05
LS-feats	**0.95**	**0.86**	**0.89**	**0.92**	**0.88**	**0.98**
NegBin	1.00	1.00	1.00	1.00	1.00	1.00

Table 2: Results for dataset EC-all. Metric values relative to NegBin (each column). ETS could not be run at this scale.

6 Conclusions. Future Work

In this paper, we developed a framework for maximum likelihood learning of probabilistic latent state forecasting models, which can be seen as principled time series extensions of generalized linear models. We pay special attention to the intermittent and bursty statistics of demand, characteristic for the vast inventories maintained by large retailers or e-commerce platforms. We show how approximate Bayesian inference techniques can be implemented in a robust and highly scalable way, so to enable a forecasting system which runs safely on hundred of thousands of items and hundreds of millions of item-days.

We can draw some conclusions from our comparative study on a range of real-world datasets. Our proposed method strongly outperforms competitors on sales data from fast and medium moving items. Besides good short term forecasts due to temporal smoothness and well-calibrated growth of uncertainty, our use of a feature vector seems most decisive for medium term forecasts. On slow moving items, simpler methods like NegBin [15] are competitive, even though they lack signal models which could be learned from data.

We are investigating several directions for future work. Our current system uses time-independent ISSMs, in particular $\boldsymbol{g}_t = [\alpha]$ means that the same amount of innovation variance is applied every day. This assumption is violated by our data, where a lot more variation happens in the weeks leading up to Christmas or before major holidays than during the rest of the year. To this end, we are exploring learning two parameters: α_h during high-variation periods, and α_l for all remaining days. We also plan to augment the state $\boldsymbol{l}_t$ by seasonality[10] factors [10, Sect. 14] (both $\boldsymbol{a}_t$, $\boldsymbol{g}_t$ depend on time then).

One of the most important future directions is to learn about and exploit dependencies between the demand time series of different items. In fact, the strategy to learn and forecast each item independently is not suitable for items with a short demand history, or for slow moving items. One approach we pursue is to couple latent processes by a shared (global) linear or non-linear function.

Acknowledgements

We would like to thank Maren Mahsereci for determining the running time figures, and the Wupper team for all the hard work without which this paper would not have happened.

[10] Currently, periodic seasonality is dealt with by features in $\boldsymbol{x}_t$.

References

[1] D. Barber. Expectation correction for smoothing in switching linear Gaussian state space models. *Journal of Machine Learning Research*, 7:2515–2540, 2006.

[2] D. Barber, T. Cemgil, and S. Chiappa. *Bayesian Time Series Models*. Cambridge University Press, 1st edition, 2011.

[3] M. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Unit, UCL, 2003.

[4] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 1st edition, 2006.

[5] G. Box, G. Jenkins, and G. Reinsel. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 4th edition, 2013.

[6] N. Chapados. Effective Bayesian modeling of groups of related count time series. In E. Xing and T. Jebara, editors, *International Conference on Machine Learning 31*, pages 1395–1403. JMLR.org, 2014.

[7] J. Durbin and S. Koopman. *Time Series Analysis by State Space Methods*. Oxford Statistical Sciences. Oxford University Press, 2nd edition, 2012.

[8] Tom Heskes and Onno Zoeter. Expectation propagation for approximate inference in dynamic Bayesian networks. In A. Darwiche and N. Friedman, editors, *Uncertainty in Artificial Intelligence 18*. Morgan Kaufmann, 2002.

[9] R. Hyndman and Y. Khandakar. Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, 26(3):1–22, 2008.

[10] R. Hyndman, A. Koehler, J. Ord, and R. Snyder. *Forecasting with Exponential Smoothing: The State Space Approach*. Springer, 1st edition, 2008.

[11] P. McCullach and J.A. Nelder. *Generalized Linear Models*. Number 37 in Monographs on Statistics and Applied Probability. Chapman & Hall, 1st edition, 1983.

[12] T. Minka. Expectation propagation for approximate Bayesian inference. In J. Breese and D. Koller, editors, *Uncertainty in Artificial Intelligence 17*. Morgan Kaufmann, 2001.

[13] H. Rue and S. Martino. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of Roy. Stat. Soc. B*, 71(2):319–392, 2009.

[14] L. Snyder and Z. Shen. *Fundamentals of Supply Chain Theory*. John Wiley & Sons, 1st edition, 2011.

[15] R. Snyder, J. Ord, and A. Beaumont. Forecasting the intermittent demand for slow-moving inventories: A modelling approach. *International Journal on Forecasting*, 28:485–496, 2012.

[16] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (NSDI)*, page 2, 2012.