Two Echelon Distribution Systems: Applications to a Luxury Goods Retailer

October 11, 2015

We have so far been concerned with the theory of multi-location multi-echelon supply chains with capacity constraints.

In this chapter, we consider a luxury goods manufacturer and retailer, and use our insights from previous chapters to develop some quantitative planning methods that may prove useful to this retailer in managing their supply chain.

Our methods, of course, require as inputs information about the supply chain in question. Using data pertaining to this luxury goods company that we were able to obtain, we develop strategies to estimate the parameters required as inputs to our quantitative planning methods.

We begin by describing the supply chain in question. We then provide a description of the data that were provided to us, together with various summary statistics pertaining to these data. We then discuss the estimation problem of using past demands to estimate future demands. Finally, we use insights from previous chapters to design some quantitative methods that may be useful in these settings.

1 Introduction

The high-level structure of our company's supply chain is as follows. The company has two large depots in North America (in California and New Jersey respectively). These depots order inventory from manufacturing plants, at high fixed cost and with lead times ranging from a few weeks to a few months, depending on the shipping method used, and supply various regional retailers in North America. Each retailer is assigned to one and only one of these depots.

Deliveries from the central depots to the retailers are contracted out to a courier service. Deliveries leave late at night or early in the morning. Some of the retailers benefit from so-called 'custom critical' (CC) shipping, and receive those deliveries before they open each morning (we

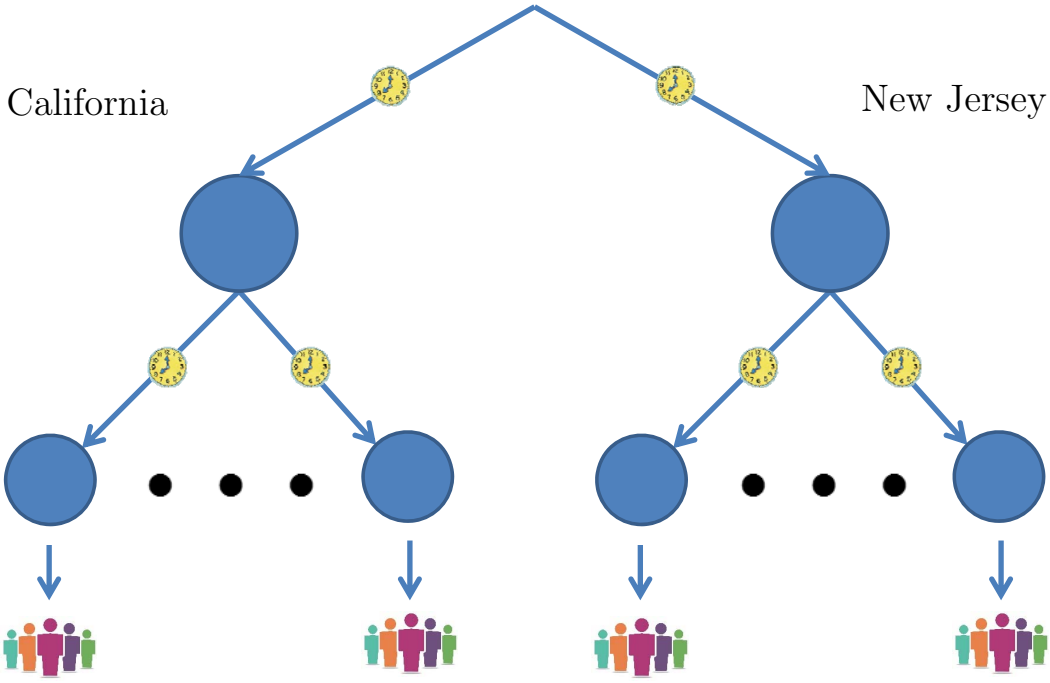

Figure 1: The supply chain at the luxury goods manufacturer studied in this thesis. The two upper circles represent the depots in California and New Jersey (which are restocked from manufacturing plants), and the lower circles represent retail locations around North America, which face stochastic demands. The small yellow clocks represent the delays ('lead times') that are experienced between orders and deliveries, both at the depot and at the individual retailers.

classify these retailers as having a lead time of zero). Other retailers only receive these shipments later in the day (we classify these retailers as having a lead time of one).

Figure 1 illustrates this supply chain diagramatically.

Our company has 132 retail outlets in North America, for which we are provided data. One of these outlets (number U120) was annotated "currently closed". The remaining outlets are partitioned as follows

- 52 retailers on the west coast. Of these
  - 25 have zero lead times, thanks to custom critical (CC) shipping.
  - 27 have one day lead-times.
- 79 retailers on the east coast. Of these

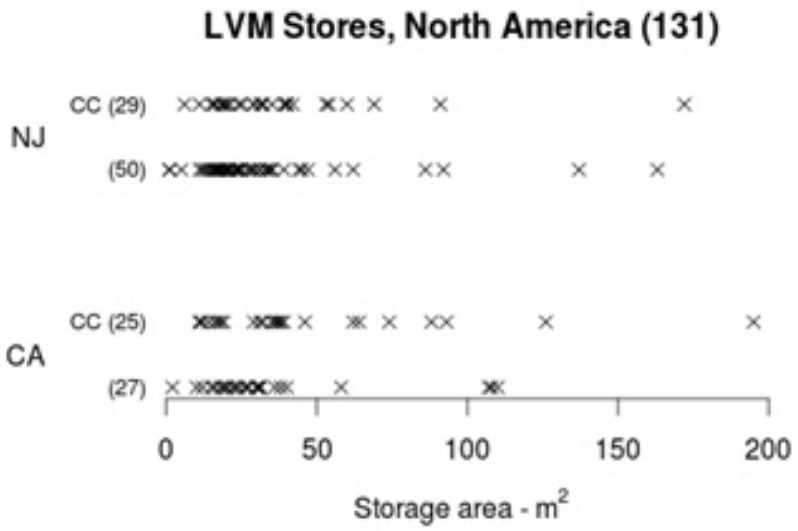

Figure 2: Store areas at the luxury goods manufacturer studied in this thesis. Stores are classified as either 'east coast' stores (NJ) or 'west coast' stores (CA), and as receiving either custom critical shipments denoted CC or normal shipments.

- 29 have zero lead times, thanks to custom critical (CC) shipping.
- 50 have one day lead-times.

We were also provided with the floor area of each retailer. These data are summarized in figure 2. Unfortunately, these floor areas are difficult to interpret because no data is provided as to the storage area occupied by each product, or as to the configuration of each storage area (i.e., whether any shelves are available, how tall they are, etc...). It is therefore difficult to know how many of each product can fit in each store given its area.

We were provided with one year's worth of sales data, and we restricted our attention to the top 106 SKUs only, which accounted for 55% of all sales.

We found that daily sales were all integer-valued and ranged from -68 units to 41 units per day. Negative sales were interpreted as returns.

- 87% of sales figures were not given; we assume that no sales were made on these days.
- 0.5% of sales were equal to 0. It is unclear exactly how these differ from the missing sales figures, but our contact at the company hypothesized that these may correspond to situations in which a sale was followed by a return on a given day.

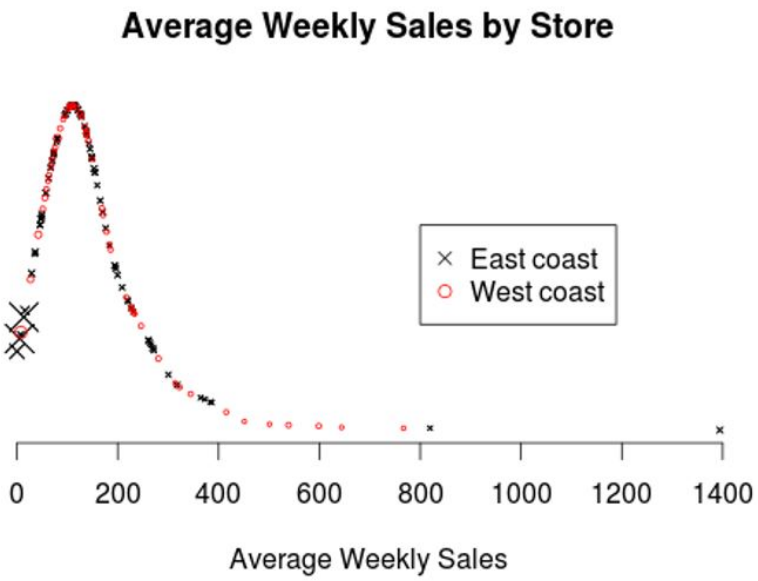

Figure 3: Summary of aggregate sales over all products at the luxury goods manufacturer studied in this thesis. Each point corresponds to a single retailer. The x position of that point represents the average weekly sales at that retailer, and the y position represents the frequency of these sales in our sample. The size of each point represents the coefficient of variation of weekly sales at that particular retailer.

- 9.29% of sales figures were equal to 1. (This corresponds to 74.37% of all sales that were not missing or equal to 0).

Our sales distributions were therefore heavily skewed, with most of their density at 0 or 1.

Figure 3 summarizes *aggregate* weekly sales over *all* products at the retailers in our sample. Given the large number of SKUs, it is clear from these data that each individual SKU averages much less than one sale per day.

1.1 Current supply chain practices

Based on various discussions with our contact at the company, our understanding is that current supply chain practices revolve around a heuristic order-up-to policy.

Every day, the inventory in each retailer is observed, and demands for the next day are predicted. A lookup table is then used that maps these two quantities to the amount that should be ordered.

2 Estimating Future Demand

Any effort to make optimal shipping decisions in this supply chain must begin with daily demand estimation for each SKU and each retailer. Indeed, without a reliable estimate of future demand at each of the retail locations, it becomes difficult to decide how much inventory should be allocated there. This estimation problem is complicated in our case by the fact daily demands are so small. Indeed, when demands are large, they can be modelled using continuous distributions – any rounding errors on large demand numbers are bound to be small. When demands are small, however, we are forced to use a discrete distribution.

There is a vast literature on demand estimation. Most papers, however, focus on consumer choice (see Guadagni and Little (1983), Cooper and Nakanishi (1988), Berry, Levinsohn, and Pakes (1995), and many more, dealing with increasing levels of complexity), but these methods seek to model market share rather than absolute demand. The literature seems to be sparse on the estimation of absolute demand, possibly because when demand is continuous (as is often assumed), the demand estimation problem reduces to a simple regression. Indeed, to our knowledge, only Bajari et al. () have very recently begun to study this class of problem using machine learning techniques, but even their work assumes continuous demands.

2.1 Picking covariates

Before we discuss the kind of statistical model most appropriate for the task of demand estimation, we discuss the choice of covariates we may use to predict future demand. These are summarized in Table 1.

Clearly, the retailer and product in question will be the two most crucial elements in estimating demand. It is unclear, however, whether these are best included in our model as covariates, or whether a different model should be fit for every store/product combination (or indeed, for every store and every product). We will experiment with each option.

2.2 Estimation model

If demands were (or could be assumed to be) normally distributed, a simple linear regression of our demands against the set of covariates discussed above could be used to find the mean and standard deviation of these distributions.

Unfortunately, when this assumption cannot be made, a linear regression can at most provide

`Store`	The store in question
`Product`	The product in question
`Day`	The day of the week (Mon-Sun)
`Month`	The month of the year
`Trend`	The number of days that have elapsed since January 1st 1900. We use this as a 'trend' indicator, to capture any systematic increase in sales across the period considered
`tM1`	The sales on the day *before* the day in question. Included to track 1-day correlations
`tM7`	The sales *seven days* before the day in question. Included to track 7-day correlations

Table 1: Potential covariates we may use to predict future demand. The first four covariates are categorical, the remaining ones are continuous.

the *mean* of each demand, rather than its full distribution. Furthermore, linear regression is likely to result in a non-integer figure for each daily demand, and rounding would result in a large error.

Looking at more sophisticated methods, two options are available to us. The first is to make some parametric assumption about the demand (for example, to assume that the demand has a zero-inflated Poisson distribution), and fit a model to estimate the parameters of this distribution. The second is to use a completely non-parametric model to estimate the distribution.

We use the non-parametric estimation route for two reasons. First, we have very little prior information to inform a choice of parametric distribution, other than knowledge of the fact each demand distribution is supported on a small set of points (as discussed above, most demands are 0, 1, or 2). Second, this small support makes a multinomial distribution particularly easy to fit to our data.

We consider two different models to fit a multinomial demand model. To describe these models, we shall need some notation. Suppose we have N observations of demand $D_1, \cdots, D_N$ with corresponding covariates $\boldsymbol{X}^{(1)}, \cdots, \boldsymbol{X}^{(N)}$, each in $\mathbb{R}^P$. Suppose further that we assume (or observe) that demands can only take values in a finite set $\{d_1, \cdots d_K\}$ of size K.

Our models are then as follows

Multinomial logistic regression is a natural extension of logistic regression to cases with multiple outcomes. We assign one parameter vector $\boldsymbol{\beta}^{(k)} \in \mathbb{R}^P$ to each possible outcome, and

let

$$\mathbb{P}(D_i = d_k) = \frac{\exp\left(\boldsymbol{\beta}^{(k)} \cdot \boldsymbol{X}^{(i)}\right)}{\sum_{\kappa=1}^{K} \exp\left(\boldsymbol{\beta}^{(\kappa)} \cdot \boldsymbol{X}^{(i)}\right)}$$

(Note that adding any fixed vector to each $\boldsymbol{\beta}$ leaves the probabilities unchanged, which means this model is not identifiable. To avoid this difficulty, we set $\boldsymbol{\beta}^{(1)} = \mathbf{0}$, and find the probability of other demands with respect to this 'baseline' demand.)

Ordered logistic regression is similar to multinomial logistic regression, but makes a stronger assumption. In particular we only estimate *one* vector $\boldsymbol{\beta} \in \mathbb{R}^P$, and a set of parameters $\{\gamma_1, \cdots, \gamma_{K-1}\}$, and set

$$\text{logit}\left(\mathbb{P}(D_i \leq d_k)\right) = \boldsymbol{\beta} \cdot \boldsymbol{X}^{(i)} - \sum_{\kappa=1}^{k} \gamma_\kappa$$

where $\text{logit}(p) = \log(\frac{p}{1-p})$, and $\gamma_K \equiv \infty$.

To understand the rationale behind this model, recall that the logit of a probability gives the *odds* of that probability. In this model, the mean of each demand depends on the covariates, but once the mean is determined, the odds of observing one demand over another is fixed and does not depend on the vector of covariates. (This explains the name 'ordered logistic regression' – the outcomes are ordered by the γ parameters).

In our specific example, this means that the overall scale of demand is determined by the covariates, but that the odds of buying two products vs. one product (for example) is fixed.

This is clearly a far more restrictive model, but it does have the advantage of requiring the estimation of far fewer parameters ($P + K - 1$ parameters v.s. $P(K-1)$ parameters for multinomial logistic regression).

2.3 Model fitting

Fitting these models should – in theory – be as simple as using the equations above to construct a likelihood and maximizing this likelihood with respect to the parameters. Indeed, packages are available that do precisely that and perform reasonably well.

When the size of the data increases, however, fitting becomes far more complicated for three reasons. First, the resulting problems are very high dimensional, especially if some of the covariates

are factors. Second, the expressions for the likelihood are not simple. And third, the scale of the data makes each step in any algorithm computationally costly.

As we shall see in later sections, the best-performing models for the data we were provided require the solution of a separate model for each SKU/retailer combination. Since each of these models require only data from one SKU at one store, they are relatively small, and the standard packages provided in R are sufficient to fit them. The algorithms in this section, therefore, are only used to initially evaluated the larger models. Nevertheless, this material is of intrinsic interest and we therefore present it here. The rest of this section, however, may be skipped without interrupting the flow of this chapter.

The algorithm we use to fit these models on large scale data is based on FISTA, introduced by Beck and Teboulle (2009), which uses a quadratic approximation to the function to be optimized, and does not require the calculation of second derivatives. Readers are directed to the paper in question for details of FISTA, and we relegate the details of calculating first derivatives of the log-likelihoods to Appendix A. We evaluate our FISTA based algorithms – both for ordered logit regression and for multinomial logistic regression – on a simulated data set containing 10,000 points and a varying number of covariates. We also run R's built-in functions on the same datasets.

2.3.1 Multinomial logistic regression

Figure 5 compares the accuracy of our FISTA-based function and of the built-in R function for multinomial logistic regression, and figure 4 compares their running times.

It is clear from both these figures that our algorithm performs better and faster on a models with a large number of covariates. When the number of covariates is small, however, the built in R function does better. Indeed, FISTA requires a set up step to approximate the Lipschitz constant of the likelihood function in question. With complicated functions like ours, this step can be particularly slow. If the model is large enough, this time is more than compensated by the fact Hessians do not need to be calculated at each step, but for small problem instances, it does slow our algorithm considerably.

2.3.2 Ordered logistic regression

Figure 7 compares the accuracy of our FISTA-based function and of the built-in R function for multinomial logistic regression, and Figure 6 compares their running times.

The situation is somewhat more complicated than that we observe for multinomial logistic

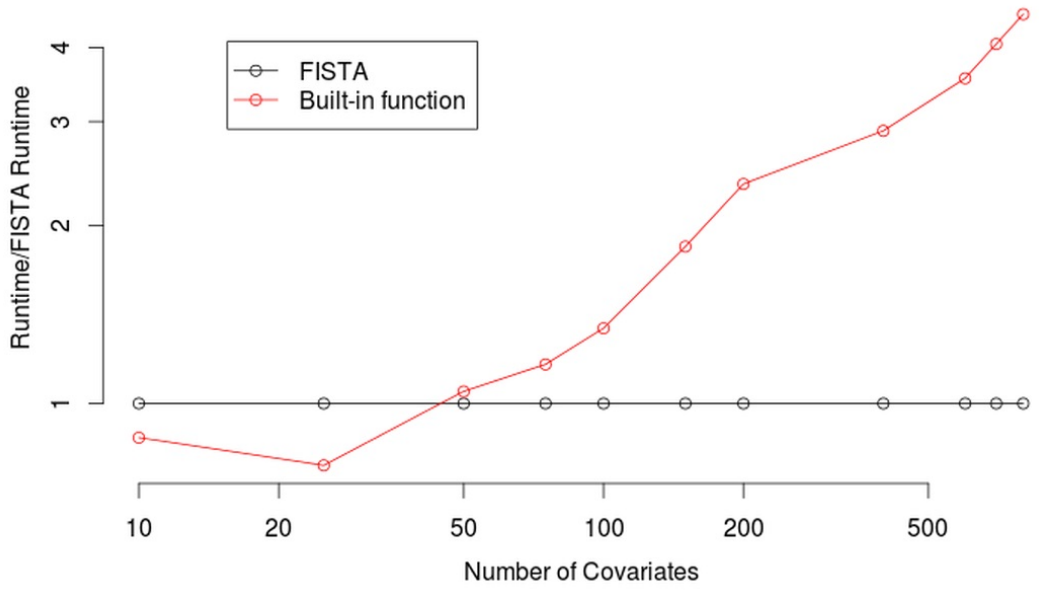

Figure 4: Runtimes for our FISTA based algorithm for multinomial logistic regression as compared to the built-in R function. The x-axis indicates the number of covariates in our artificial model, and the y-axis indicates running time as a fraction of the time taken by our FISTA algorithm. All models were run on 10,000 data points. Note that the FISTA time quoted only accounts for the FISTA iterations themselves. Because our FISTA algorithm is written in C++ and the data is generated in R, some work is required to prepare the data for processing. Because the R function, which works directly on the R data, does not need to perform these steps, their runtime is not included above.

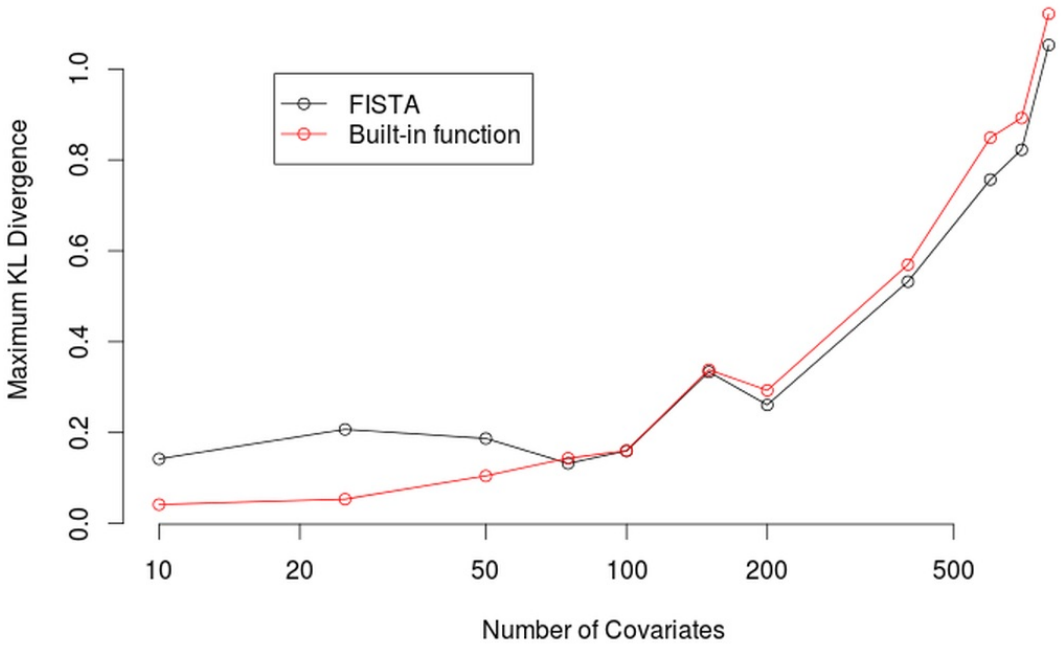

Figure 5: Accuracy of our FISTA based algorithm for multinomial logistic regression as compared to that of the built-in R version. The x-axis indicates the number of covariates in our artificial model, and the y-axis indicates the maxium KL-divergence (over all points in our sample) between predicted outcome probabilities and actual outcome probabilities. All models were run on 10,000 data points.

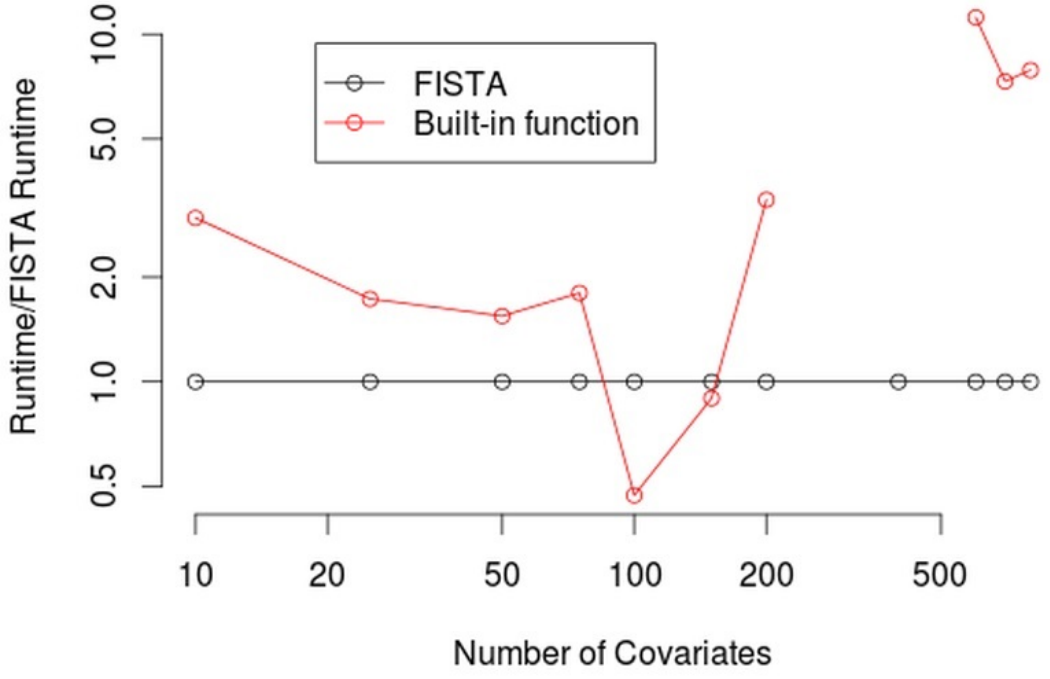

Figure 6: Runtimes for our FISTA based algorithm for ordered logistic regression as compared to the built-in R version. The x-axis indicates the number of covariates in our artificial model, and the y-axis indicates running time in seconds as a fraction of FISTA times. All models were run on 10,000 data points. Note that the FISTA time quoted only accounts for the FISTA iterations themselves. Because our FISTA algorithm is written in C++ and the data is generated in R, some work is required to prepare the data for processing. Because the R function, which works directly on the R data, does not need to perform these steps, their runtime is not included above. Note that the point at 400 covariates is missing because the R function was unable to converge for this dataset.

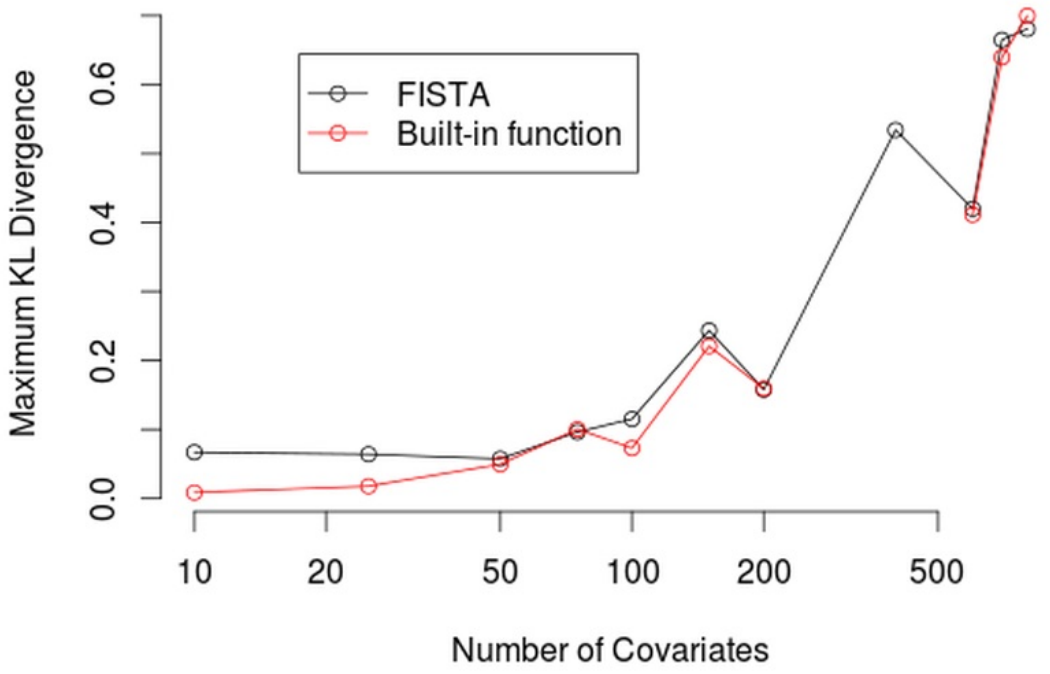

Figure 7: Accuracy of our FISTA based algorithm for ordered logistic regression as compared to the built-in R version. The x-axis indicates the number of covariates in our artificial model, and the y-axis indicates the maxium KL-divergence (over all points in our sample) between predicted outcome probabilities and actual outcome probabilities. All models were run on 10,000 data points. Note that the point at 400 covariates is missing because the R function was unable to converge for this dataset.

regression. Indeed, whilst our algorithm performs very accurately, and faster than the R version, there is a small number of models (at 100 and 150 covariates) for which the R version converges significantly faster. Furthermore, there is one model (with 400 covariates) at which the built-in R function is not able to converge at all, and returns no results.

This is not surprising. As discussed in Appendix A, the computational details involved in evaluating the ordered logistic regression likelihood and its derivatives are significantly more complicated than those for multinomial logistic regression (indeed, as is discussed at length in Appendix A, a number of modifications to FISTA had to be made to ensure rapid convergence). It therefore follows that the corresponding algorithms will display greater instability.

2.4 Evaluating Model Performance

We shall be fitting four kinds of models. The first will include all our data, with SKU and stores as covariates. At the other extreme, we shall be fitting a different model for each SKU/Store combination. Finally, we shall fit one set of models for each SKU, and one set of models for each store. In this section, we devise some measures that can be used to evaluate the performance of each of these models, and decide which yields the best results.

In a simple linear regression, model evaluation is comparatively simple. An in- or out-of- sample R^2 or mean squared error can be calculated and used as a measure of the model's accuracy. With binary classification, ROC curves can be plotted.

Unfortunately, model performance is far harder to assess in a multinomial setting. For a given set of covariates, a multinomial fit provides a set of probabilities for each outcome $\{d_1, \cdots, d_k\}$, and it is unclear how we should asses how 'accurate' these outputs are.

The two most obvious options are a deviance-based measure, and a generalized χ-squared test.

Multinomial logistic regression is an example of a generalized linear model. As in all generalized linear models, a deviance-based goodness-of-fit test could therefore be used. In ordinary least squares models, the 'sum of squared residuals' is often used as a measure of fit. The deviance is a generalization of this measure to models that produce a fit by maximizing a more general likelihood. Further details are available in texts on Generalized Linear Models (McCullagh and Nelder (1989) is the seminal reference). Unfortunately, the deviance works best when comparing two versions of the same model using a different selection of covariates. In our case, however, we need to compare entirely different models, each fit on different datasets.

A second tempting measure is a variation on the *chi*-squared test adapted to multinomial

distributions, developed by Fagerland, Hosmer, and Bofin (2008). This tests first groups data into a number of clusters, based on the probability of observing the first outcome d_1 (in other words, the first cluster contains points with low probability of observing d_1, the second cluster points with slightly higher probability of observing d_1, etc...). A contingency table is then constructed, listing the predicted count of each outcome in each cluster, together with the real count, and a simple χ-squared test is run on this contingency table. Unfortunately, for this test to be statistically significant, a minimum number of observations is required in each cell of the χ-squared contingency table produced in the test, which makes it unusable in this instance, since some outcomes ('-1' demand in particular) are extremely sparse.

Instead, we suggest the following four measures of performance:

'Count measure' (CM) : Look at the outcome with the highest predicted probability, and compare it to the *actual* outcome. Find the proportion of points for which these two outcomes do not match. We shall call this the 'count measure' (CM).

The higher the CM, the worse the fit.

'Probability measure' (PM) : For each point i in our data set, look at the predicted probability vector, and consider the probability corresponding to the *actual* outcome D_i for that point – call this probability p. We then assign to this point a score of $-p\log(p)$. We find the average of these scores over all points, and call it the 'probability measure' (PM).

The higher the PM, the worse the fit.

This measure is motivated by the KL-divergence. Indeed, this score is precisely the KL-divergence between our predicted probability vector and a vector of zeros containing a '1' in the position corresponding to the actual outcome.

These two measures are intuitive, and will be useful in deciding which models to pick. However, there are likely to paint a very dreary picture – indeed, with so few covariates available, it is unlikely the model will *exactly* predict the right outcome with such high probability.

Instead, therefore, we develop two new measures that we call 'rolling measures'. Recall that each of our points represents demand at a given store for a given product on a given day. Our new measures consider a rolling window of 40 days, and looks at the performance of the model *in aggregate* over that window. The results over each rolling window are then averaged.

'Rolling Count Measure' (RCM) Our first measure simply looks at the *total demand* forecast

	CM	PM	RCM	RPM
Full	0.53	0.30	0.25	0.11
By SKU	0.51	0.29	0.14	0.06
By Store	0.51	0.30	0.20	0.09
Individual Models	0.46	0.28	0.07	0.02

Table 2: Results from four multinomial logistic regression models. The first row corresponds to a model in which a model is run on all 11 stores and 11 SKUs, with the store and SKU as covariates. The last row corresponds to average measures over models in which each SKU/Store combination has a single model. The middle two rows correspond to average measures over models including either only one SKU (but multiple stores) or vice versa. The four measures (CM, PM, RCM, RPM) are described in the main body of the text.

in the window, and compares it to the *actual demand* observed in that window. We use the average absolute difference per day in our window, averaged over all windows, as our RCM.

'Rolling Probability Measure' (RPM) Our second measure tests the accuracy of the split among the different outcomes. For each window, we find the predicted proportion of observations from each outcome. We then compare these proportions to the actual proportions observed using the KL-divergence. The average such KL-divergence is averaged over all windows.

2.5 Results

We fit the four models described in the previous section, and calculate the four measures described above for each of these models. In all cases, ordered logistic regression performs very poorly; clearly, the additional assumption in the ordered logistic regression model is too restrictive.

We therefore use multinomial logistic regressions, and our results are listed in Table 2, and illustrated in Figure 8. Note that in all cases, we used the same data for fitting and evaluation. Whilst we would ideally have split our data into a training and test set, the large numbers of parameters to estimate and the relatively small number of data points available (especially for the individual models) required the use of all data available for fitting.

These results make it very clear that by all measures, individual models for each SKU and store combination perform better than any combined model. It is interesting to note that in general, SKU models perform better than store models. This implies that there is far more variability in demand across different SKUs than across different stores, a result that is not particularly surprising.

It is also heartening to note that these models tend to perform very well indeed. The CM

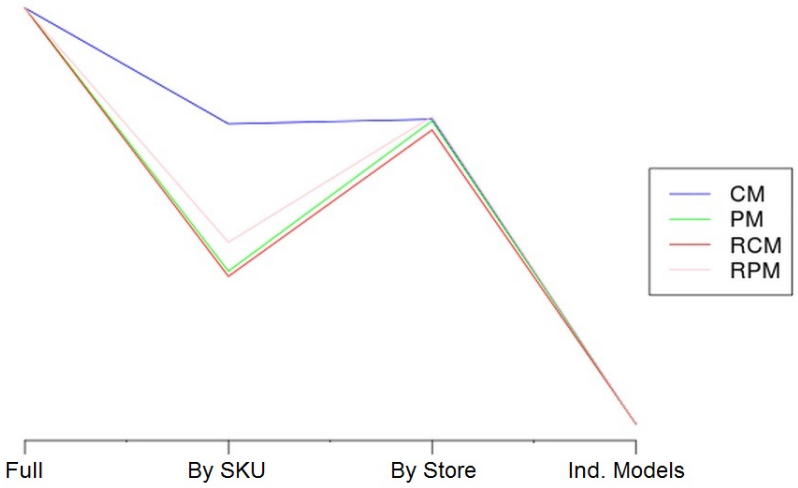

Figure 8: Diagramatic representation of the data in table 2.

measure (which we expected to perform poorly) correctly classifies 54% of points, and the rolling count measure implies that every day, the demand is miss-estimated by only 0.07 units on average.

3 Optimizing the Supply Chain

Having discussed methods to obtain reliable estimates of future demand, we are ready to consider methods for optimizing the supply chain itself.

Ideally, we would use the methods studied in the previous chapters and apply them to this specific supply chain. Regrettably, this would yield poor results in the current setting. Indeed, because most demands observed are very small, the resulting mean demand is tiny in most cases, and the resulting coefficient of variation is very large. For coefficients of variation as large as these, the gaps observed between the upper and lower bounds discussed in the previous chapters are very large.

We propose, instead, a simplification of the models discussed in the previous chapters to this particular supply chain. We first note that because the leadtimes to the depot in this case are much longer than those to the individual retailers (sometimes by an order of magnitude), and because fixed order costs to the depot are so large (orders have to be placed from manufacturing plants and production initiated), the depot will always have abudant inventory to supply each of the retailers. This means that we do not need to concern ourselves with optimizing the *entire* supply chain – it is enough to simply consider each retailer individually. Having determined the optimal allocation

for each retailer, we can be sure that the depot will always be able to supply as much inventory as needed to fulfill this optimal need. This, effectively, reduces our two-echelon multi-location problem to a single-echelon, single-location problem, albeit one with capacity constraints.

We now formulate a mathematical program – similar to the heuristics discussed in the previous chapters – that will allow us to determine the optimal allocation for each store under such a scheme. We then use simulation to compare the performance of this heuristic to other, more naive, heuristics.

3.1 Finding the optimal allocation

All notation in this section will echo the notation in our handling of the multi-location, multi-product case. In particular, recall the following definitions:

- $u^i_{j,(t\to\tau)}$ denotes the sum of all product-i demands at retailer j from time t to time $\tau + \ell_j$:

$$u^i_{j,(t\to\tau)} = \sum_{\tau'=t}^{\tau+\ell_j} u^i_{j,(\tau')}$$

  and $\mu^i_{j,(t\to\tau)} = \mathbb{E}\left(u^i_{j,(t\to\tau)}\right)$

  Note, therefore, that $\mu^i_{j,(t\to t)} \equiv \tilde{\mu}^i_{j,(t)}$, and $u^i_{j,(t\to t)} \equiv \tilde{u}^i_{j,(t)}$.

- We let

$$Q^i_{j,(t\to\tau)}(X) = p^i_{j,(\tau+\ell_j)}\mathbb{E}\left[X - u^i_{j,(t\to\tau)}\right]^- + h^i_{j,(\tau+\ell_j)}\mathbb{E}\left[X - u^i_{j,(t\to\tau)}\right]^+$$

Let us first consider the myopic problem at store j. Suppose we are in period t and that we place orders $\left\{\bar{w}^i_{j,(t)}\right\}_{i=1}^I$ – these orders will arrive in period $t + \ell_j$, and the expected holding and backorder costs in that period will be given by $\sum_{i=1}^I Q^i_{j,(t\to t)}(x^i_{j,(t)} + \bar{w}^i_{j,(t)})$.

Assuming we take this myopic view, therefore, we can find the optimal orders $\left\{\bar{w}^i_{j,(t)}\right\}_{i=1}^I$ by solving the following optimization problem

$$\begin{aligned}
\max \quad & \sum_{i=1}^I Q^i_{j,(t\to t)}\left(x^i_{j,(t)} + \bar{w}^i_{j,(t)}\right) \\
\text{s.t.} \quad & \bar{w}^i_{j,(t)} \geq 0 && \forall i \\
& \sum_{i=1}^I \max\left(x^i_{j,(t)} + \bar{w}^i_{j,(t)} - u^{i,[\alpha_j]}_{j,(t\to t)}, 0\right) \leq \chi_{j,(t+\ell_j)}
\end{aligned}$$

The first constraint insists that all deliveries to the store be positive. The second requires that the current inventory in the store, plus any deliveries, does not exceed the capacity with a certain probability.

As we have seen in previous chapters, this sort of myopic thinking often results in results that are suboptimal.

We did, however, also see that looking a few periods into the future can result in far better results.

As in previous chapter, we therefore consider a less myopic approach, that considers several periods in the future when deciding on optimal allocations for this period. Specifically, suppose we look κ^i periods forward for product i, and let $\bar{w}^i_{j,(\tau)}$ denote the allocation of product i to our store in period τ. Our non-myopic problem then becomes

$$\begin{aligned}
\max \quad & \sum_{i=1}^{I}\sum_{\tau=t}^{t+\kappa^i} Q^i_{j,(t\to\tau)}\left(x^i_{j,(t)} + \textstyle\sum_{\tau'=t}^{\tau}\bar{w}^i_{j,(\tau')}\right) & \\
\text{s.t.} \quad & \bar{w}^i_{j,(\tau)} \geq 0 & \forall i,\tau \\
& \sum_{i=1}^{I}\max\left(x^i_{j,(t)} + \sum_{\tau'=t}^{\tau}\bar{w}^i_{j,(\tau')} - u^{i,[\alpha_j]}_{j,(t\to\tau)}, 0\right) \leq \chi_{j,(\tau+\ell_j)} & \forall\tau
\end{aligned}$$

Note that because each the κ^i may be different for different products, the capacity constraints for later periods will not necessarily include every product. Because we will only be using this program to decide on allocations in period t, however, the fact later capacity constraints may not be comprehensive will not affect the validity of our results.

Finally, we note that in this application, the demand distributions are discrete, and $\tilde{u}^i_{j,(t\to\tau)}$ is supported on a finite set of consecutive integers $\{-L,\cdots,U\}$. Each of the Q are then piece-wise linear and convex, and the optimization program above becomes

$$\begin{aligned}
\max \quad & \sum_{i=1}^{I}\sum_{\tau=t}^{t+k^i}\Theta^i_{j,\tau} & \\
\text{s.t.} \quad & \bar{w}^i_{j,(\tau)} \geq 0 & \forall i,\tau \\
& C^i_{j,(\tau)} \geq x^i_{j,(t)} + \textstyle\sum_{\tau'=t}^{\tau}\bar{w}^i_{j,(\tau')} - u^{i,[\alpha_j]}_{j,(t\to\tau)} & \forall i,\tau \\
& C^i_{j,(\tau)} \geq 0 & \forall i,\tau \\
& \textstyle\sum_{i=1}^{I} C^i_{j,(\tau)} \leq \chi_{j,(\tau+\ell_j)} & \forall\tau \\
& \Theta^i_{j,\tau} & \\
& \quad \geq p^i_{j,(\tau+\ell_j)}\mu^i_{j,(t\to\tau)} + \Big[(p^i_{j,(\tau+\ell_j)} + h^i_{j,(\tau+\ell_j)})F^i_{j,(t\to\tau)}(u) & \\
& \qquad -p^i_{j,(\tau+\ell)}\Big]\left(x^i_{j,(t)} + \textstyle\sum_{\tau'=t}^{\tau}\bar{w}^i_{j,(\tau')}\right) & \\
& \qquad\qquad +(p^i_{j,(\tau+\ell_j)} + h^i_{j,(\tau+\ell_j)}) & \\
& \qquad\qquad\quad \Big[\textstyle\sum_{u'=-L}^{u-1} F^i_{j,(t\to\tau)}(u') - uF^i_{j,(t\to\tau)}(u)\Big] & \forall i,\tau,u
\end{aligned}$$

This is a linear program, solvable using a number of standard packages.

3.2 Performance

Ideally, we would test the strategy suggested above on the data provided by our luxury goods retailer, and compare the performance of this strategy with the methodology currently in use there. Unfortunately, this is difficult to do for two reasons. (1) We do not have access to meaningful data about store capacities; we have data pertaining to the area of each store, bu no data bout the size of each product, or about the shelving configuration at each of these stores. (2) For most of the time period our data pertains to, we are only given sales data, not inventory data, at each of the stores. It is therefore difficult to know what ordering strategy was used.

As a result, we devise some other basic heuristic policies, and benchmark the strategy above against these heuristics. It is important for our benchmarking heuristics to ensure the capacity at each retailer is not exceeded. Indeed, with no capacity constraints, a simple (s, S) policy is proveably optimal, and would by definition perform better than our suggested strategy above.

These are the four allocation strategies we consider:

Non-myopic constrained algorithm : this is the optimal algorithm described in the previous section. It looks at four periods into the future for each product and implicitly includes capacity constraints in its formulation.

Myopic constrained algorithm : this algorithm is identical to our optimal algorithm, but it only considers *one* period into the future instead of four.

Non-myopic unconstrained algorithm : this algorithm is identical to our optimal algorithm, but it does *not* include any capacity constraints. As a result, the optimal allocations as prescribed by this algorithm – if blindly followed – will result in severe capacity overflows.

To (at least partially) mitigate this problem, we correct the output of this algorithm at each step by ensuring that the sum of all inventory delivered is equal to the sum of all inventory delivered under the optimal algorithm. We do this by randomly subtracting delivery amounts for randomly chosen products, until the total allocation is small enough.

The correction method is designed to replicate what may happen when such a policy is implemented. Presumably, when inventory arrives that consistently exceeds available capacity, excess inventory is returned or discarded as it arrives.

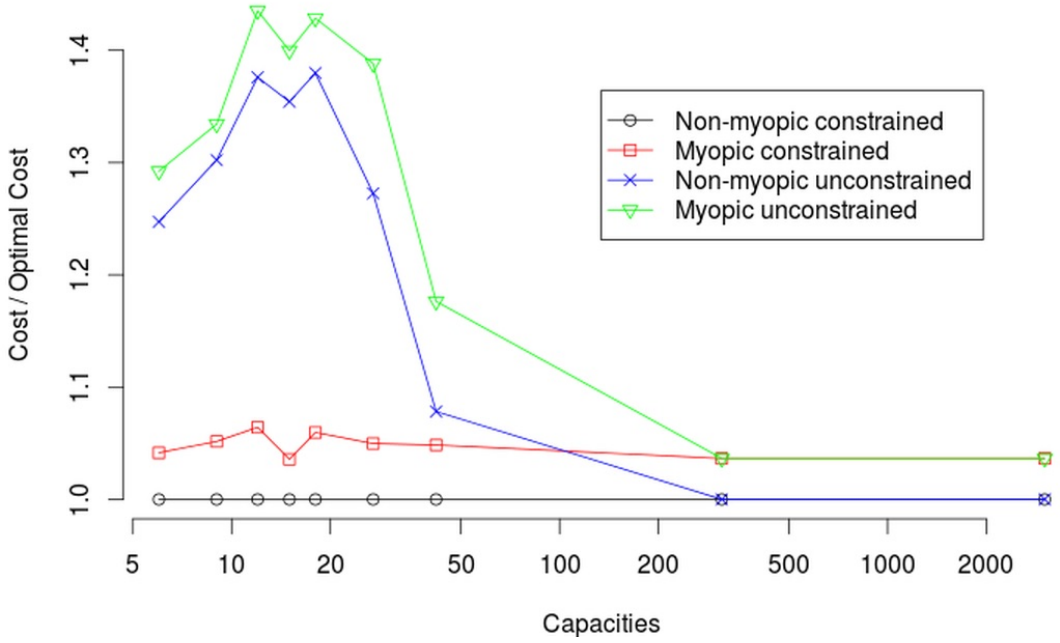

Figure 9: Performance of our non-myopic allocation algorithm compared to heuristics. The x-axis corresponds to the capacity of the retailer and the y axis gives the average cost per period, as a multiple of the cost of our optimal algorithm.

Myopic unconstrained algorithm : this algorithm only looks one period ahead, and does not include any capacity constraint. A similar fix is applied to ensure the capacity constraints are not too severely violated.

We simulate each of these algorithms on a sample problem with 10,000 periods, stationary demands and non-stationary holding and backorder costs. We use 10 products and a lead-time of 3. The total aggregate demand for all products in each period is 12, with a standard deviation of 3. We run simulations with capacities equal to the mean demand, one and two standard deviations below the mean demand, and one, two, five, 10, 100 and 1000 standard deviations above the mean demand (the last simulation was designed to model a situation in which capacity constraints are not binding). We consider the average cost per period using each of the algorithms above.

The results of these simulations are illustrated in figures 9 and 10. Some salient points:

- Our optimal algorithm clearly performs much better than any of the heuristics.

  Not only are the optimal costs better under our algorithm than under the heuristics, but the probability of a capacity overflow is also lower. In other words, despite the fact our algorithm

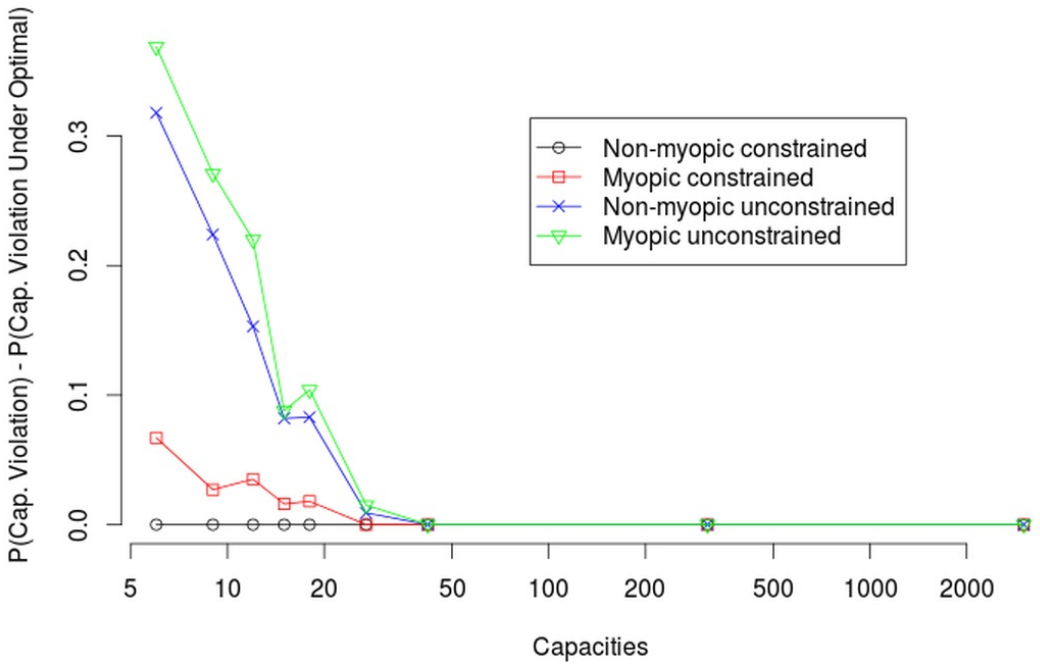

Figure 10: Capacity constraint performance of our non-myopic allocation algorithm compared to heuristics. The x-axis corresponds to the capacity of the store in question and the y axis gives the probability of the capacity at the store being exceeded minus that probability when applying our optimal algorithm.

does a better job at staying within capacity, it also performs at a lower cost.

- It is interesting to note that the cost-advantage of using our algorithm over the heuristics does not monotonically decrease with capacity. This could be because at extremely low capacities, all algorithms are bound to do poorly regardless – as such, the improved performance of our algorithm is less pronounced.

- It is surprising that our heuristic algorithms result in more capacity violations than the optimal algorithm. Indeed, recall that the heuristic policy constrains the total deliveries to be identical to those in the optimal algorithm; one might therefore expect a similar number of violations in both cases.

  The difference arises because backordered items are not counted as 'extra space' in our retailer. In other words, if the capacity is 10 and the retailer contains 15 units of one item and -5 of another (ie: 5 backordered units), the capacity is *still* violated. Our optimal algorithm knows this, and assigns inventories bearing this fact in mind. Our heuristic algorithm does not.

- It is interesting to note that moving from a non-myopic to a myopic algorithm worsens performance, but not by much. This may be because we are considering a situation in which the demands are stationary – only the holding and backorder costs vary from period to period. There is therefore not as much to lose by using a non-myopic policy.

- As the capacity of our store increases to infinity, both non-myopic algorithms converge to the same performance, as may be expected. The myopic algorithms (whether constrained or unconstrained) also converge to the same cost, though that cost is slightly higher than that of the non-myopic algorithms, as may be expected.

- As the capacity of our store increases to infinity, the probability of capacity violation converges to zero for all our algorithms, as would be expected.

4 An Interactive System

The results in the previous section are promising. Unfortunately, they could be difficult to implement in practice in a business environment. Indeed, they require the solution of a non-trivial linear program – this could be challenging for non-technical business executives. In a high-pressure, fast-moving environment, methods such as these are unlikely to be adopted unless they can seamlessly

be integrated into existing workflows.

To make our method easier to integrate into a typical workflow, therefore, we construct a browser-based system that takes infrastructure data (including holding and backorder costs) and demand distributions at each retailer, and automatically calculates the optimal allocation to each retailer using the strategy above by running a server-side R-Script.

The system is best experienced 'live', but some screen shots are included in figure 11.

5 Conclusion

We were provided with demand data for a luxury goods manufacturer. We experimented with multinomial logistic regression as a method for estimating future demand, with a number of different model choices.

Having estimated future demand, we considered methods for optimal allocations in such distribution systems. Unfortunately, the methods discussed in our previous chapters are not suitable in this case, where the coefficients of variation of the demand distributions are very large. We noted, however, that the likely abundance of inventory at the depot made it possible to consider each retailer individually. We constructed an linear program to solve the optimal allocation problem for each of these retailers, and showed that it performed significantly better than heuristic policies that could have been employed to artificially meet the capacity constraints in our system, both in terms of optimal cost and in terms of capacity violations.

Finally, we constructed an interactive online system suitable for use by non-technical executives. The system allows the input of holding and backorder costs as well as demand distributions, and returns optimal allocations to the retailer in question.

References

Bajari, P. et al. "Demand Estimation with Machine Learning and Model Combination". https://www.utexas.edu/co
2-4.pdf, retrieved May 6th, 2015.

Beck, A. and M. Teboulle (2009). "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems". In: *SIAM J. Imaging Sciences* 2.1, pp. 183–202.

Berry, S., J. Levinsohn, and A. Pakes (1995). "Automobile Prices in Market Equilibrium". In: *Econometrica* 63.4, pp. 841–890.

Cooper, L. G. and M. Nakanishi (1988). *Market Share Analysis*. Kluwer Academic Publisher, Boston, MA.

Fagerland, M. W., D. W. Hosmer, and A. M. Bofin (2008). "Multinomial goodness-of-fit tests for logistic regression models". In: *Statistics in Medicine* 27, pp. 4238–4253.

Guadagni, P. M. and J. D. C. Little (1983). "A Logit Model of Brand Choice Calibrated on Scanner Data". English. In: *Marketing Science* 2.3. ISSN: 07322399. URL: `http://www.jstor.org/stable/184043`.

McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models*. Chapman and Hall/CRC.

A Using FISTA to fit Multinomial Logistic and log-Logistic Models

In this section, we consider the practical details of estimating the models in section 2 using FISTA. In particular, we will show that the resulting log-likelihood is concave, discuss numerical issues involved in its calculation, and calculate its derivatives.

With one exception that will be discussed later, we used the standard FISTA algorithm for convex optimization, as devleoped by Beck and Teboulle (2009). For this reason, we will not give a detailed account of our algorithm – readers are directed to that paper for details.

We shall assume throughout this appendix that there are N observations of demand $\{D_i\}_{i=1}^N$, each with a corresponding set of covariates $\left\{\boldsymbol{X}^{(i)}\right\}_{i=1}^N$. We assume that each demand D_i can take one value in the set $\{d_1, \cdots, d_k\}$.

B Multinomial logistic regression

In multinomial logistic regression, we assume that each demand is distributed as follows

$$\log\left(\frac{\mathbb{P}(D_i = d_k)}{\mathbb{P}(D_i = d_1)}\right) = \boldsymbol{\beta}^{(k)} \cdot \boldsymbol{X}^{(i)}$$

where $\{\boldsymbol{\beta}^{(2)}, \cdots, \boldsymbol{\beta}^{(k)}\}$ are vectors in $\mathbb{R}^p$ to be estimated. For notational convenience, we shall define a vector $\boldsymbol{\beta}^{(1)} \equiv \mathbf{0}$.

This immediately gives, for every k,

$$\mathbb{P}(D_i = d_k) = \frac{\exp(\boldsymbol{\beta}^{(k)} \cdot \boldsymbol{X}^{(i)}}{\sum_{\kappa=1}^K \exp(\boldsymbol{x}^{(\kappa)} \cdot \boldsymbol{X}^{(i)}}$$

and

$$\log \mathbb{P}(D_i = d_k) = \boldsymbol{\beta}^{(k)} \cdot \boldsymbol{X}^{(i)} - \log\left(\sum_{\kappa=1}^K \exp(\boldsymbol{\beta}^{(k)} \cdot \boldsymbol{X}^{(i)})\right)$$

B.1 Concavity of the log-likelihood

The log-likelihood is trivially concave by the concavity of the log-sum-exp function.

B.2 Evaluating the log-likelihood

Only the second term may cause a problem when evaluating the log likelihood. Indeed, if one of the exponential terms inside the logarithm is extremely large, the entire expression could overflow.

To solve this problem, we re-write the second term as follows

$$\begin{aligned}\log\left(\sum_{\kappa=1}^{K}\exp(\boldsymbol{\beta}^{(k)}\cdot\boldsymbol{X}^{(i)})\right) &= \max_k(\boldsymbol{\beta}^{(k)}\cdot\boldsymbol{X}^{(i)}) \\ &\quad + \log\left(\sum_{\kappa=1}^{K}\exp\left[\boldsymbol{\beta}^{(k)}\cdot\boldsymbol{X}^{(i)} - \max_k(\boldsymbol{\beta}^{(k)}\cdot\boldsymbol{X}^{(i)})\right]\right)\end{aligned}$$

This ensures that every term in the logarithm is reasonably sized, and makes the evaluation of this expression far more numerically stable.

B.3 Derivatives of the log-likelihood

We define a set of matrices $\left\{\mathrm{E}^{(1)},\cdots,\mathrm{E}^{(k)}\right\}$, each in $\mathbb{R}^{(K-1)P\times}$, as follows

$$\mathrm{E}^{(1)} = 0_{(K-1)P\times P}$$

$$\mathrm{E}^{(k)} = \begin{pmatrix} 0_{(k-1)P\times P} \\ I_{P\times P} \\ 0_{(K-k)P\times P}\end{pmatrix}$$

We then have that

$$\nabla_{\boldsymbol{\beta}}\log\mathbb{P}(D_i=d_k) = \mathrm{E}^{(k)}\boldsymbol{X}^{(i)} - \frac{\sum_{\kappa=1}^{K}\exp(\boldsymbol{\beta}^{(\kappa)}\cdot\boldsymbol{X}^{(i)})\mathrm{E}^{(\kappa)}\boldsymbol{X}^{(i)}}{\sum_{\kappa=1}^{K}\exp(\boldsymbol{\beta}^{(\kappa)}\cdot\boldsymbol{X}^{(i)})}$$

C Ordered Logistic Regression

In ordered logistic regression, we assume that each demand is distributed as follows

$$\begin{aligned}\operatorname{logit}\left[\mathbb{P}(D_i\le d_k)\right] &= \ln\left(\frac{\mathbb{P}(D_i\le d_k)}{1-\mathbb{P}(D_i\le d_k)}\right) \\ &= \left(\sum_{\kappa=1}^{k}\gamma_\kappa\right) - \boldsymbol{\beta}\cdot\boldsymbol{X}^{(i)} \\ &\equiv -\zeta_k^i\end{aligned}$$

Where $\boldsymbol{\beta}$ and $\{\gamma_1, \cdots, \gamma_{K-1}\}$ are parameters to be estimated, and we assume $\gamma_K = \infty$. We constrain γ to be strictly positive. This immediately gives

$$\mathbb{P}(D_i \leq d_k) = \frac{1}{1+\exp(\zeta_k^i)}$$

and

$$\mathbb{P}(D_i = d_k) = \begin{cases} \dfrac{1}{1+\exp(\zeta_1^i)} & k = 1 \\ \dfrac{1}{1+\exp(\zeta_k^i)} - \dfrac{1}{1+\exp(\zeta_{k-1}^i)} & 1 < k < K \\ 1 - \dfrac{1}{1+\exp(\zeta_{K-1}^i)} & k = K \end{cases}$$

Simplifying, multiplying the second line by $\exp(-\zeta_k^i)$ and remembering that $\zeta_{k-1}^i = \gamma_k + \zeta_k^i$, we find that

$$\mathbb{P}(D_i = d_k) = \begin{cases} \dfrac{1}{1+\exp(\zeta_1^i)} & k = 1 \\ \dfrac{\exp(\gamma_k) - 1}{1 + \exp(-\zeta_k^i) + \exp(\gamma_k)\left[1+\exp(\zeta_k^i)\right]} & 1 < k < K \\ \dfrac{1}{1+\exp(-\zeta_{K-1}^i)} & k = K \end{cases}$$

Taking logarithms

$$\ln \mathbb{P}(D_i = d_k) = \begin{cases} -\ln\left[1+\exp(\zeta_1^k)\right] & k = 1 \\ \ln\left(\exp[\gamma_k] - 1\right) & \\ \quad -\ln\left[1 + \exp(-\zeta_k^i) + \exp(\gamma_k)\left[1+\exp(\gamma_k^i)\right]\right] & 1 < k < K \\ -\ln\left[1+\exp(-\zeta_{K-1}^i)\right] & k = K \end{cases}$$

C.1 Log-concavity of the Likelihood

Before we consider any algorithm to optimize the likelihood, we will show it is log-concave. Before we prove this, let us prove the following lemma.

Lemma 1. *The function*

$$f(a,b) = \ln\left(1 + e^{-a} + e^{b} + e^{a+b}\right)$$

is jointly convex in a and b.

Proof. First, consider that

$$\nabla f(a,b) = \frac{1}{1+e^{-a}+e^{b}+e^{a+b}} \begin{pmatrix} e^{a+b} - e^{-a} \\ e^{a+b} + e^{b} \end{pmatrix}$$

and

$$\nabla^2 f(a,b) = \frac{1}{\left(1+e^{-a}+e^{b}+e^{a+b}\right)^2} D$$

where

$$D_{1,1} = e^{a+b} + e^{-a} + e^{a+2b} + e^{b-a} + 4e^{b}$$

$$D_{1,2} = D_{2,1} = e^{a+b} + 2e^{b} + e^{b-a}$$

$$D_{2,2} = e^{a+b} + 2e^{b} + e^{b-a} + 4e^{a+2b} + 2e^{2b} + 2e^{2a+2b}$$

Setting $K = \left(1+e^{-a}+e^{b}+e^{a+b}\right)$, we find that

$$\begin{aligned}
\frac{\left|\nabla^2 f(a,b)\right|}{K} &= a^{2a+2b} + 2e^{a+2b} + e^{2b} + 4e^{2a+3b} + 2e^{a+3b} + 2e^{3a+3b} \\
&\quad + e^{b} + 2e^{b-a} + e^{b-2a} + 4e^{2b} + 2e^{2b-a} + 2e^{a+2b} \\
&\quad + e^{2a+3b} + 2e^{a+3b} + e^{3b} + 4e^{2a+4b} + 2e^{a+4b} + 2e^{3a+4b} \\
&\quad + e^{2b} + 2e^{2b-a} + e^{2b-2a} + 4e^{3b} + 2e^{3b-a} + 2e^{a+3b} \\
&\quad + 4e^{a+2b} + 8e^{2b} + 4e^{2b-a} + 16e^{a+3b} + 8e^{3b} + 8e^{2a+3b} \\
&\quad - e^{2a+2b} - 2e^{a+2b} - e^{2b} \\
&\quad - 2e^{a+2b} - 4e^{2b} - 2e^{2b-a} \\
&\quad - e^{2b} - 2e^{2b-a} - e^{2b-2a}
\end{aligned}$$

simplifying

$$\begin{aligned}
\frac{\left|\nabla^2 f(a,b)\right|}{K} &= 4e^{2a+3b} + 2e^{a+3b} + 2e^{3a+3b} \\
&\quad + e^{b} + 2e^{b-a} + e^{b-2a} + 4e^{2b} \\
&\quad + e^{2a+3b} + 2e^{a+3b} + e^{3b} + 4e^{2a+4b} + 2e^{a+4b} + 2e^{3a+4b} \\
&\quad + e^{2b} + 4e^{3b} + 2e^{3b-a} + 2e^{a+3b} \\
&\quad + 4e^{a+2b} + 3e^{2b} + 4e^{2b-a} + 16e^{a+3b} + 8e^{3b} + 8e^{2a+3b}
\end{aligned}$$

This is clearly positive. Thus, $f(a,b)$ is jointly convex in a and b. □

We are now ready to show that the likelihood is log-concave.

Proposition 1. *The likelihood above is log-concave.*

Proof. Consider that we can write the log-likelihood as

$$\ln \mathbb{P}(D_i = d_k) = \begin{cases} -f(-\zeta_1^i, -\infty) & k = 1 \\ \ln\left(e^{\gamma_k} - 1\right) - f(\zeta_k^i, \gamma_k) & 1 < k < K \\ -f(\zeta_{K-1}^i, \infty) & k = K \end{cases}$$

where f is defined as in Lemma 1.

Now, the first part of the second term can trivially be shown to be convex. Then, convexity of f and convexity of affine combination and sums immediately shows that the likelihood is concave. □

C.2 Evaluating the log-likelihood

When any of the ζ_k^i are very large, the log likelihood can be difficult to evaluate numerically. In this section, we consider the numerical evaluation of this log-likelihood in such cases.

In the discussion that follows, we let ϵ represent a large number – say 50.

We now consider each term in the log-likelihood separately

- $\boxed{k = 1}$ In this case, if $\zeta_1^i \geq \epsilon$, we can simply set

$$-\ln\left[1 + \exp(\zeta_1^i)\right] = -\zeta_1^i$$

- $\boxed{k = K}$ Similar considerations apply.

- $\boxed{1 < k < K}$ Both terms in this part pose difficulties

  - For the first term, if γ_k is very small, we can write

$$\ln\left(e^{\gamma_k} - 1\right) = \ln(\gamma_k)$$

  - For the second term, consider two cases

    * If $\gamma_k + \zeta_k^i < 0$, we can write

$$\ln\left[1 + \exp\left(-\zeta_k^i\right) + \exp\left(\gamma_k\right)\left[1 + \exp\left(\zeta_k^i\right)\right]\right] = -\zeta_k^i + \ln\left[1 + \exp(\zeta_k^i) + exp\left(\gamma_k + \zeta_k^i\right)\left[1 + \exp\left(\zeta_k^i\right)\right]\right]$$

      In this case, since $\gamma_k > 0$, we are guaranteed to have $\zeta_k^i < 0$. As such, every exponentiated term above is small, and evaluation poses no numerical problem.

* If $\gamma_k + \zeta_k^i > 0$, we can write

$$\ln\left[1 + \exp\left(-\zeta_k^i\right) + \exp\left(\gamma_k\right)\left[1 + \exp\left(\zeta_k^i\right)\right]\right]$$
$$= \gamma_k + \ln\left[1 + \exp\left(-\gamma_k\right) + \exp\left(-\zeta_k^i - \gamma_k\right) + \exp\left(\zeta_k^i\right)\right]$$

If $\zeta_k^i > \epsilon$, then this entire expression reduces to $\gamma_k + \zeta_k^i$. If not, every exponentiated term is guaranteed to be small, and evaluation poses no numerical difficulties.

C.3 Derivatives of the log-Likelihood

We now consider the computation and evaluation of the derivatives of the log-likelihood.

To do this, we first note that $\nabla_{\boldsymbol{\beta}} \zeta_k^i = \boldsymbol{X}^{(i)}$. Having said that, algebraic manipulation yields

$$\nabla_{\boldsymbol{\beta}} \ln \mathbb{P}(D_i = d_k) = \boldsymbol{X}^{(i)} \begin{cases} -\frac{1}{1+\exp(-\zeta_1^i)} & k = 1 \\ \frac{1-\exp(\gamma_k + 2\zeta_k^i)}{1+\exp(\zeta_k^i)+\exp(\gamma_k+\zeta_k^i)+\exp(\gamma_k+2\zeta_k^i)} & 1 < k < K \\ \frac{1}{1+\exp(\zeta_{K-1}^i)} & k = K \end{cases}$$

The derivatives with respect to $\boldsymbol{\gamma}$ are considerably more complicated. Letting $\boldsymbol{e}^{(\kappa)}$ denote a vector of zeroes in $\mathbb{R}^{K-1}$ with a single 1 in the κth position, and $\boldsymbol{e}^{(1\to k)} = \sum_{\kappa=1}^{k} \mathbb{E}^{(\kappa)}$, we have that $\nabla_{\boldsymbol{\gamma}} \zeta_k^i = -\boldsymbol{e}^{(1\to k)}$. We then obtain

$$\nabla_{\boldsymbol{\gamma}} \ln \mathbb{P}(D_i = d_k) = \begin{cases} \frac{1}{1+\exp(-\zeta_1^i)} \boldsymbol{e}^{(1)} & k = 1 \\ -\frac{1-\exp(\gamma_k+2\zeta_k^i)}{1+\exp(\zeta_k^i)+\exp(\gamma_k+\zeta_k^i)+\exp(\gamma_k+2\zeta_k^i)} \boldsymbol{e}^{(1\to k)} & \\ \quad -\frac{1+\exp(\zeta_k^i)}{1+\exp(-[\zeta_k^i+\gamma_k])+\exp(-\gamma_k)+\exp(\zeta_k^i)} \boldsymbol{e}^{(k)} & 1 < k < K \\ \quad +\frac{1}{1-\exp(-\gamma_k)} \boldsymbol{e}^{(k)} & \\ -\frac{1}{1+\exp(\zeta_{K-1}^i)} \boldsymbol{e}^{(1\to K-1)} & k = K \end{cases}$$

C.4 Evaluating the derivatives

Evaluating the derivatives will also cause some numerical problems. Let us consider each part of the derivative.

C.4.1 Derivatives with respect to β

The $k = 1$ and $k = K$ terms will not pose any numerical problems. For the $1 < k < K$ term, the denominator is always ≥ 1 and so we need to worry about dividing by zero. We do, however, need

to worry about a situation in which both the numerator and denominator get very large. This will only occur if $\gamma_k + 2\zeta_k^i < \epsilon$ (once again, we let ϵ represent a large number – say 50). We deal with this situation as follows

- If $\zeta_k^i > \epsilon$, then $\exp(\gamma_k + 2\zeta_k^i)$ will dominate the denominator, and set the entire expression to -1.

- Otherwise, γ_k must be large, and so $\exp(\zeta_k^i)$ will be negligible compared to terms involving γ_k. As such, set

$$\begin{aligned} \frac{1 - \exp(\gamma_k + 2\zeta_k^i)}{1 + \exp(\zeta_k^i) + \exp(\gamma_k + \zeta_k^i) + \exp(\gamma_k + 2\zeta_k^i)} &\approx \frac{\exp(\gamma_k + 2\zeta_k^i)}{\exp(\gamma_k + \zeta_k^i) + \exp(\gamma_k + 2\zeta_k^i)} \\ &= -\frac{1}{1 + \exp(-\zeta_k^i)} \end{aligned}$$

C.4.2 Derivatives with respect to γ

Once again, the $k = 1$ and $k = K$ terms will not pose any numerical problems. For the $1 < k < K$ term, we note that the denominators are always ≥ 0, so we need not worry about dividing by zero. Once again, however, we need to worry about each of the first two terms to ensure *both* the numerator and denominator don't get too large

First term : problems arise if $\gamma_k + 2\zeta_k^i \geq \epsilon$. In that case

- If $\zeta_k^i > \epsilon$, then $\exp(\gamma_k + 2\zeta_k^i)$ will dominate everything, and we can write

$$-\frac{1 - \exp(\gamma_k + 2\zeta_k^i)}{1 + \exp(\zeta_k^i) + \exp(\gamma_k + \zeta_k^i) + \exp(\gamma_k + 2\zeta_k^i)} \approx 1$$

- If not, γ_k must be very large, and any terms not involving it with vanish. We can therefore write

$$-\frac{1 - \exp(\gamma_k + 2\zeta_k^i)}{1 + \exp(\zeta_k^i) + \exp(\gamma_k + \zeta_k^i) + \exp(\gamma_k + 2\zeta_k^i)} \approx \frac{1}{1 + \exp(\zeta_k^i)}$$

Second term : problems arise if $\zeta_k^i > \epsilon$. Furthermore, recalling that $\gamma_k > 0$, this would also make the second and third term in the denominator very small. We can therefore say

$$-\frac{1 + \exp(\zeta_k^i)}{1 + \exp(-[zeta_k^i + \gamma_k]) + \exp(-\gamma_k) + \exp(\zeta_k^i)} \approx -1$$

C.5 Modifying FISTA

One last details remains to be discussed. Whilst FISTA is a very reliable method for general convex optimization, it unfortunately fails to converge quickly enough on multinomial logistic regression.

The reason is that this particular log-likelihood involves two very different kinds of variables. The $\boldsymbol{\beta}$ variables on the one hand, and the $\boldsymbol{\gamma}$ variables on the other.

Standard FISTA uses a symmetric quadratic to approximate the function to be optimized, with a single curvature constant L. This forces the same curvature to be used in the $\boldsymbol{\beta}$ dimensions and in the $\boldsymbol{\gamma}$ dimensions, which results in poor convergence.

To mitigate this problem, we adopt an approach in which a few preliminary steps of the algorithm are carried out with a single L value, at which point a re-calibrating step is carried out, in which the dimensions corresponding to $\boldsymbol{\gamma}$ are given a *different* L value than those corresponding to $\boldsymbol{\beta}$. The algorithm then proceeds as normal.

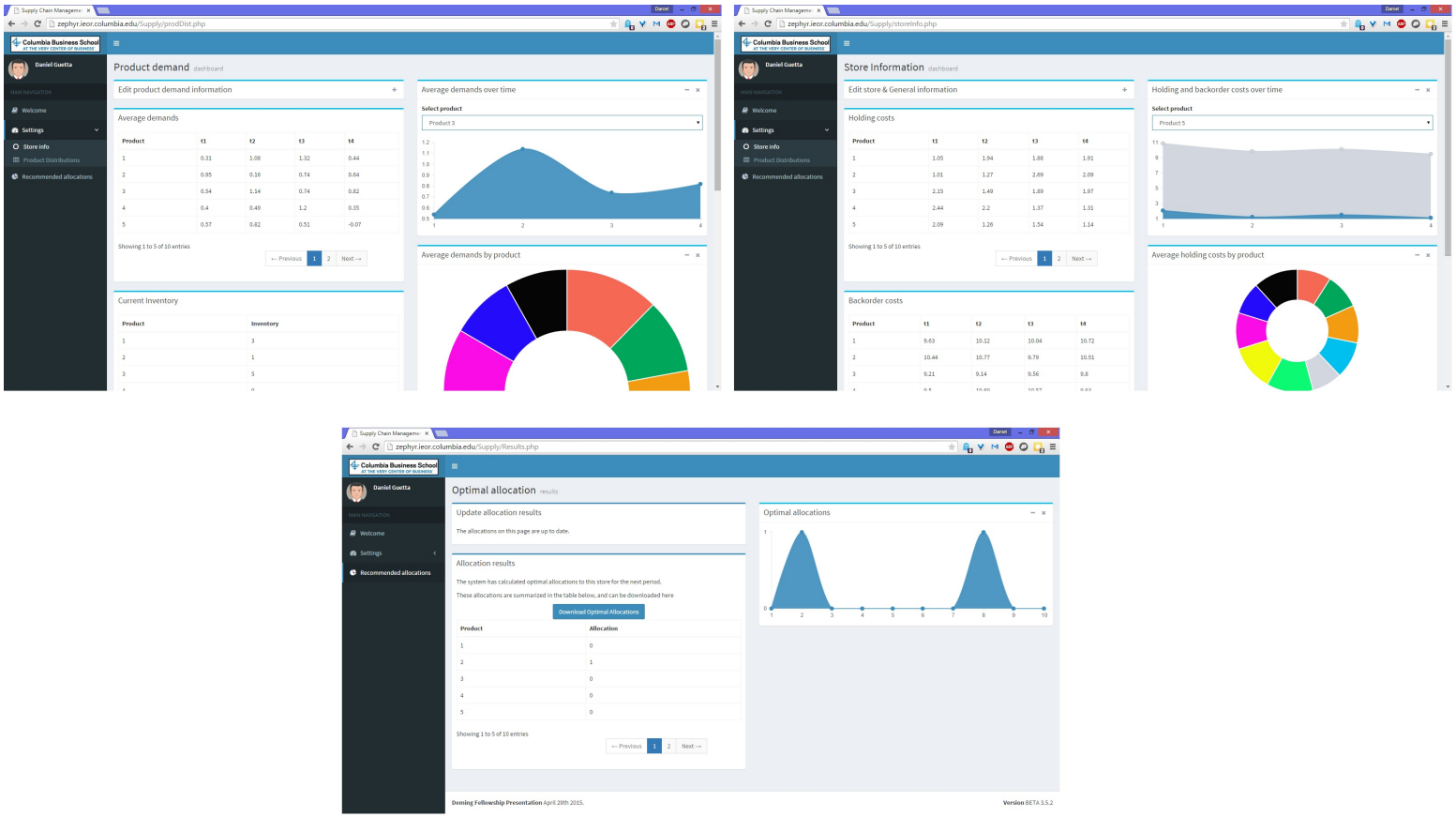

Figure 11: Screenshot of the interactive allocation system we designed. (Top left) An interface to allow the upload of demand characteristics for the system. (Top right) An interface to allow the upload of infrastructure data, including holding and backorder costs. (Bottom) An interface to launch the R script to determine allocations, displays these allocations, and makes them available for download.

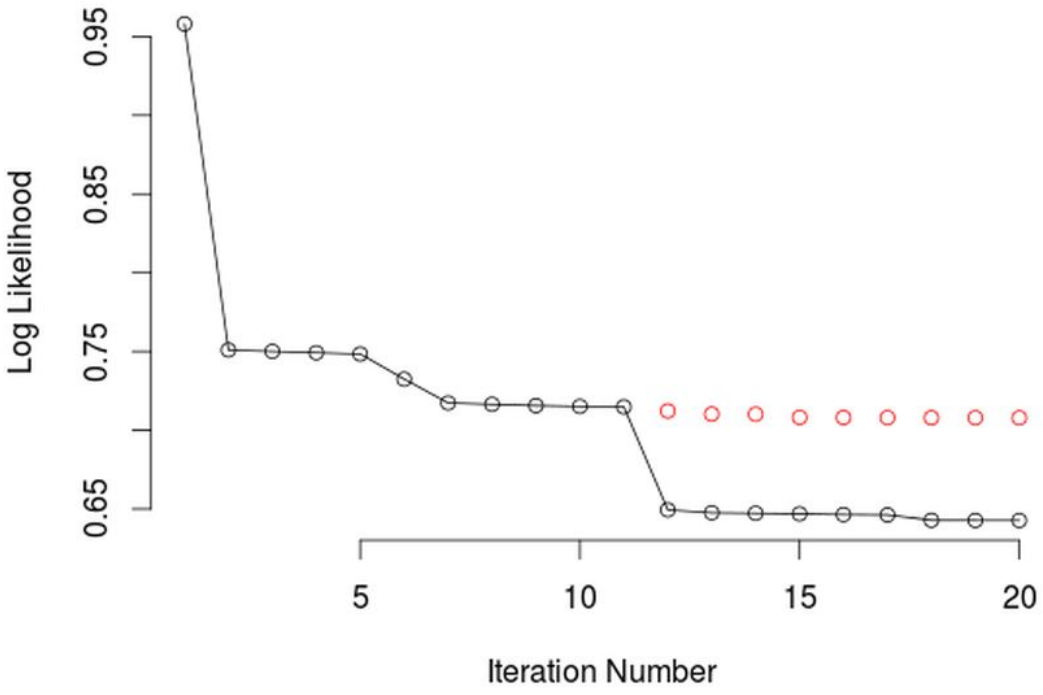

Figure 12: This diagram illustrates the effect of using a FISTA algorithm with two different L values – one for the $\boldsymbol{\beta}$ dimensions, and one for the $\boldsymbol{\gamma}$ dimensions. Both algorithms use a single value of L for 12 iterations, and then diverge. The algorithm in red continues using a single value of L for all dimensions. The algorithm plotted in black uses two different values of L and clearly converges faster.