LARGE-SCALE HOTEL DEMAND PREDICTION WITH EM

John Gao
Smith School of Business
Queen's University
john.gao@queensu.ca

May 13, 2019

ABSTRACT

This report introduces a faster implementation of the EM algorithm used in van Ryzin et al. (2017). The new implementation is built on top of C++, and uses the open source solver IPOPT for the nonlinear optimization component. The new implementation provides significant speed upgrades, allowing it to be used on much larger problem sizes whilst maintaining similar accuracy.

1 Introduction

Revenue management is a subfield of Operations Research which focuses on maximizing the revenue gained from a limited supply of products. Estimation of future demand acts to support the task of revenue management. The estimated demand data is then used in other revenue management models, the performance of which depends heavily on the quality of estimates.

One such method for demand estimation is the Expectation-Maximization(EM) Method, which has accuracy comparable to other state-of-the-art demand estimation methods, and superior speed. This paper proposes an improvement on the implementation of the EM method which further improves model processing speed and allows it to be used on larger, more realistic datasets.

2 Background and Related Work

2.1 Choice Based Demand Estimation

The framework used in this paper for demand estimation was first proposed by Mahajan and van Ryzin(2001), and consists of treating each customer as an ordering of product alternatives. Each customer will purchase his highest-ranked available product, which could be no product if the non-purchase alternative ranks higher than all available products. The number of possible types is factorial w.r.t. the number of product alternatives, but this can be narrowed down to a smaller set through market research and heuristics. The probability that a given customer falls into a certain customer type is predicted using the EM algorithm.

In addition to the customer's type probability, another variable that needs to be estimated is the arrival rate. An arrival is defined as a customer which looks at the available product alternatives, and then decides to either make a purchase or not. In a given time period, if there is no product purchased, then there may or may not have been a customer arrival. This probability is also predicted using the EM algorithm.

2.2 The Expectation-Maximization Algorithm

First proposed by Dempster et al. (1977), the EM algorithm is a general method used to find maximum likelihood parameters in a problem with hidden or missing data. The EM algorithm has no closed form and is an iterative method, with each iteration consisting of an expectation step and a maximization step.

The expectation step serves to estimate the full dataset, including missing data, using current parameter estimates and observed data. The maximization step then maximizes the log-likelihood of the estimated full dataset by finding new parameters. The maximization step is done using an open-source commercial-grade solver named IPOPT. This method provably converges to a global optimum parameter set with respect to the true underlying dataset in Borman (2004).

2.3 IPOPT and CPPAD

IPOPT is an open-source software library for nonlinear-optimization problems, which can be found at `https://coin-or.github.io/Ipopt/`, and implements an interior point method outlined in Wächter (2002). We use a version of IPOPT interfaced in the programming language C++. As of writing, IPOPT is the fastest open-source nonlinear solver, which is the reason that it was used in this paper.

IPOPT requires the calculation of nontrivial derivatives, which is handled by an algorithmic differentiation package named CPPAD (`https://coin-or.github.io/Ipopt/`). CPPAD generates an optimization problem compatible with IPOPT using standard mathematical notation as an input. IPOPT subsequently solves the problem given to it by CPPAD, and produces optimized decision variables.

3 Methodology

The model used in this paper follows the same theoretical framework as the one outlined in van Ryzin et al. (2017). This framework is summarized below.

Notation

T	Total number of purchase periods
t	A single purchase period, $t \in \{1, ..., T\}$
$\mathscr{P}$	Set of periods where there was a purchase
$\bar{\mathscr{P}}$	Set of periods where there was no purchase
S_t	A set of available products in period t
N	Total number of customer types
$\sigma^{(i)}$	A preference list defining a single customer type $i, i \in \{1, ..., N\}$
$\sigma^{(i)}(j_t)$	The preference rank of item j_t for customer type i. Lower numbers signify higher preference.
j_t	Number representing the product which was purchased in period t. $j_t = 0$ means non-purchase.
$\mathscr{M}_t$	The set of types which are possible in period t, given observed purchasing behavior and product availability.
r_{it}	The expected probability of customer type i in period t
m_i	The expected count of occurrences of customer type i in the dataset
x_i	Probability of customer type i over the entire dataset
λ	Customer arrival rate
a_t	Expected arrival rate in period t

The first piece of input data is transaction data, which consists of a total of T purchase periods. Each period t consists of at most one purchase j_t, which is selected from the available product set in that period S_t. j_t can be zero, in which case the customer decided not to purchase anything, or there was no customer arrival at all.

The second piece of input data is a list of product availability for each time period corresponding to the transaction data.

The third and final piece of input data is a list of N preference lists, each representing a customer type. Because the actual number of preference lists is enormous, they are cut down ahead of time using heuristics and/or market research. If there is a customer arrival in period t, the customer will compare the available items in S_t with his preference list $\sigma^{(i)}$. The customer will then choose the highest ranked available item and purchase it. If his non-purchase option outranks all available items, then he won't make any purchase.

The customer type cannot be gleaned directly from transaction data, and estimating the customer type is one of the goals of the EM algorithm. Before feeding the transaction type into the EM algorithm, the list of possible customer types in each period is narrowed further into the compatible type set $\mathscr{M}_t(j_t, S_t)$. This set is built by comparing the actual purchase in a period with what's available, and eliminating customer types which don't fit the data for that specific period. Formally:

$$\mathscr{M}_t(j_t, S_t) = \{i : \sigma^{(i)}(j_t) < \sigma^{(i)}(k) \; \forall k \in S_t, k \neq j_t\} \tag{1}$$

The EM algorithm outputs estimates for two parameters: λ and x_i, where $i \in \{1, ..., N\}$. λ is used to represent the probability that a 0 value in the transaction data represents a customer who arrived but did not make a purchase, rather than the lack of customer arrival. The probability of some new customer being in type i is represented by the variable x_i.

To summarize, there are three inputs to the algorithm; the first is a transaction vector of length T, where each j_t represents the product number which was purchased in period t. The second input is a $T \times K$ sized availability matrix, where K is the total number of product alternatives. Each row of this matrix, denoted S_t, is a vector which shows which products are available in time period t. The final input is the customer type matrix, which is size $N \times K + 1$. Each row represents a possible customer type, and each column represents a product alternative, with one of the columns representing the non-purchase option.

At this point, model implementation changes depending on whether or not non-purchases are identifiable from non-arrivals.

3.1 Uncensored Model Implementation

If we can assume that we have a dataset in which we can identify non-purchases from non-arrivals, then we can implement the uncensored model. Non-purchases will have a zero value in the transaction data, and non-arrivals will be cut out of the transaction data entirely. This eschews the need to estimate an arrival rate λ, which is set to a value of 1 in this case.

The customer type probabilities are estimated by maximizing the following likelihood function:

$$\mathscr{L}_U = \sum_{i=1}^{N} m_i \log x_i \tag{2}$$

Since x_i's are probabilities, they all must be greater than or equal to zero. Furthermore, since customer type probabilities need to sum to 1, the x variables are subject to the following constraint:

$$\sum_{i=1}^{N} x_i = 1 \tag{3}$$

The following algorithm uses all the inputs listed in the previous section, and is similar to the one used in van Ryzin et al. (2017), with changes in the maximization step. The stopping criterion used was the maximum change in any single x_i; the algorithm was set to run until this maximum change went below 1e-3. Note that since the algorithm has very fast convergence, initialization of variables and stopping criteria doesn't have a large impact in the practical sense.

Algorithm 1: Uncensored EM Algorithm

```
Data: Transaction data, customer type list, availability data
Result: Customer type probabilities $\{x_0, ..., x_N\}$
[Initialization]
Build $\mathscr{M}_t(j_t, S_t)$ sets using equation (1);
Set $\{x_0, ..., x_N\}$ to 1;
while Max change $x_i > 1e-3$ do
    Set $m_i := 0, r_{it} := 0$ for all $i = 1, ..., N, t = 1, ..., T$;
    [E-Step]:
    [Compute expected customer type probabilities]
    for $t = 1$ to $T$ do
        for $i \in M_t$ do
            $r_{it} := x_i \sum_{h \in \mathscr{M}_t(j_t, S_t)} x_h$;
        end
    end
    [Compute expected customer type arrival counts]
    for $i = 1$ to $N$ do
        for $t = 1$ to $T$ do
            $m_i += r_{it}$;
        end
    end
    [M-Step]:
    [Maximize log-likelihood of expected dataset]
    max $\sum_{i=1}^{N} m_i \log x_i$
    s.t. $\sum_{i=1}^{N} x_i = 1, x_i > 0\ \forall i$;
end
```

3.2 Censored Model Implementation

If one is unable to differentiate between non-purchases and non-arrivals, the EM algorithm would then need to predict an arrival rate λ in order to estimate how much of the non-purchases were actually non-arrivals. This only needs to be done where there is ambiguity between non-arrivals and non-purchases. For instance, if a period has no purchases but also no compatible types which would purchase nothing, then there is certainly no arrival. The lambda estimation is supported by the a_t variable which tracks predicted arrival rate in each time period. a_t is then used to weight the $r_i t$ values when calculating m_i, in order to account for customers that may or may not have been arrivals.

The final likelihood function is also different, as λ now needs to be predicted along with x_i. The new likelihood function is the following:

$$\mathscr{L}_C = \sum_{i=1}^{N} m_i \log x_i + (|\mathscr{P}| + \sum_{t \in \bar{\mathscr{P}}} a_t) \log \lambda + (|\bar{\mathscr{P}}| - \sum_{t \in \bar{\mathscr{P}}} a_t) \log(1 - \lambda) \qquad (4)$$

This function is to be maximized in the uncensored EM algorithm, while satisfying the same constraints in (3) and the further constraint that $0 \leq \lambda \leq 1$. Note that as in the uncensored algorithm, the initialization of x_i and λ are arbitrary.

Algorithm 2: Censored EM Algorithm

```
Data: Transaction data, customer type list, availability data
Result: Customer type probabilities $\{x_0, ..., x_N\}$, arrival rate $\lambda$
[Initialization]
Build $\mathscr{M}_t(j_t, S_t)$ sets using equation (1);
Set $\{x_0, ..., x_N\}$ to $\frac{1}{N}$;
Set $\lambda$ to 0.5;
while Max change $x_i > 1e-3$ do
    Set $m_i := 0, p_{it} := 0$ for all $i = 1, ..., N, t = 1, ..., T$;
    [E-Step]:
    for $t = 1$ to $T$ do
        [Compute expected customer type probabilities];
        for $i \in M_t$ do
            $p_{it} := x_i \sum_{h \in \mathscr{M}_t} x_h$;
        end
        [Compute expected customer arrival rates];
        if $j_t$ != 0 then
            [Certain arrival]
            $a_t := 1$;
        else
            if $\mathscr{M}_t(0, S_t) = \varnothing$ then
                [Certain nonarrival]
                $a_t := 0$;
            else
                [Ambiguous, calculate expected arrival probability]
                $a_t := \lambda \sum_{i \in \mathscr{M}_t(0,S_t)} x_i / (\lambda \sum_{i \in \mathscr{M}_t(0,S_t)} x_i + (1 - \lambda))$
            end
        end
    end
    [Compute expected customer type arrival counts] for $i = 1$ to $N$ do
        for $t = 1$ to $T$ do
            $m_i \mathrel{+}= a_t r_{it}$;
        end
    end
    [M-Step]:
    [Maximize log-likelihood of expected dataset]
    $\max \sum_{i=1}^{N} m_i \log x_i + (|\mathscr{P}| + \sum_{t \in \bar{\mathscr{P}}} a_t) \log \lambda + (|\bar{\mathscr{P}}| - \sum_{t \in \bar{\mathscr{P}}} a_t) \log(1 - \lambda)$
    s.t. $\sum_{i=1}^{N} x_i = 1, x_i > 0 \; \forall i, 0 \leq \lambda \leq 1$;
end
```

4 Comparison of Results

The following is a comparison of computation times on simulated datasets of varying sizes. Each size of dataset was randomly generated 30 times, and the average of those computations is provided in the table below. Parameter recovery accuracy between this model and van Ryzin et al. (2017) are comparable.

Table 1: Mean computation times for a single instance of censored demand case, in a market with 15 products and 10 customer types, in seconds.

	$\lambda = 0.2$		$\lambda = 0.8$	
T	Van Ryzin EM	C++ IPOPT EM	Van Ryzin EM	C++ IPOPT EM
10000	0.317	0.183	0.226	0.177
50000	1.600	0.514	1.624	0.526
100000	3.118	0.916	2.680	1.015

The systems running the models are similar in both cases, both with Intel Core i7 processors and 8GB of RAM. The main difference lies in programming language (van Ryzin et al. (2017) used Matlab whereas we use C++) and solving method (van Ryzin et al. (2017) use a closed form maximizer, whereas we use a more general iterative solver). Nonetheless, we see a significant improvement in processing time.

5 Discussion and Future Work

Demand estimation is but a small part of the overall revenue management pipeline, and the results of the EM process must be fed into other models.

One way to make use of accurate demand estimates is to augment the task of auxiliary value prediction. Hotels generally benefit from being able to predict auxiliary customer spend (e.g. drinks, food, etc.), in order to make more revenue by protecting rooms for high-spending customers. The usual method for this is to run some type of regression with auxiliary spend as the dependent variable, and various independent variables including stay length, previous spend, arrival date, etc.

The problem with auxiliary spend prediction is that any dataset will inherently oversample customers who book earlier relative to their stay date. This is because as time goes on, rooms will get booked, and so customers who book closer to their stay date have a higher chance of non-purchase due to lack of availability. This skews customer traits in favor of customers who book early, which introduces a bias into the auxiliary value prediction.

A possible solution to this is to replace late booker non-purchases with estimated purchase behavior to counter early booker oversampling. The EM algorithm allows us to do exactly this, by sampling customer types using the estimated type probabilities to determine estimated purchasing behavior. The frequency at which one replaces non-purchase data is determined by arrival rate λ. This would then balance out the customer dataset, which should, in theory, increase auxiliary prediction accuracy across all customers.

6 Conclusion

This report outlines a computationally faster way of implementing the EM algorithm for demand estimation based on the method used in van Ryzin et al.(2017). It also highlights possible applications of such estimates.

References

[1] Mahajan S, van Ryzin G (2001a) Inventory competition under inventory competition under dynamic consumer choice. *Operations Research* 49(5): 646–657.

[2] Dempster A, Laird N, Rubin D (1977) Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* B 39(1):1–38.

[3] Borman S (2004) The Expectation Maximization Algorithm: a Short Tutorial. Manuscript.

[4] Wächter A (2002) An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA.

[5] van Ryzin G, Vulcano G (2017) Technical Note: an Expectation-Maximization Method to Estimate a Rank-based Choice Model of Demand. *Operations Research* 65(2): 396–407.