Technical Memorandum Number 827

The Multiple-Family ELSP with Fill Rates

by

S.M. Karalli
College of Business Administration, California State University
karallis@csus.edu

A. D. Flowers
Weatherhead School of Management, Case Western Reserve University,
dale.flowers@case.edu

September 2010

Department of Operations
Weatherhead School of Management
Case Western Reserve University
330 Peter B. Lewis Building
Cleveland, Ohio 44106
The Multiple-Family ELSP with Fill Rates

S.M. Karalli∗
College of Business Administration, California State University, Sacramento,
6000 J Street, Sacramento, CA 95819
Email: karallis@csus.edu

A.D. Flowers
Department of Operations, Weatherhead School of Management,
Case Western Reserve University, 10900 Euclid Avenue, Cleveland, OH 44106
Email: dale.flowers@case.edu

September 13, 2010

Abstract

We revisit the Multiple Family Economic Lot scheduling Problem with safety stocks (MFELSP-SS) [1]. In this technical note we focus our attention on safety stock maintained to meet fill rate requirements, again considering them explicitly in the formulation.

Differences between service level and fill rate criteria are discussed. A solution procedure for the MFELSP with fill rates (MFELSP-FR) is presented for this model.

Subject classifications:
inventory/production: approximations/heuristics, multi-item, multifamily, safety stock;
production/scheduling: applications, approximations/heuristics, cyclic schedule

Area of review: Manufacturing, Service, Supply Chain Operations

*Corresponding author
1 Background

Consider an item, whose demand is normally distributed with mean $d = 1000$ units per period and standard deviation of demand $\sigma = 400$ units per period, is scheduled for production every $t$ periods. Safety stock must be maintained in order to meet a predetermined service criteria. Figure 1 depicts the behavior of safety stock as a function of time when held (a) to hedge against stockouts, with service level requirements of $SL = 95\%$, $SL = 97\%$, and $SL = 98\%$ (dashed curves), and (b) to meet fill-rate requirements of $f = 95\%$, $f = 97\%$, and $f = 98\%$ (solid curves).

**Service Level Safety Stock.** Safety stock held to hedge against stockouts varies with time according to the expression $400z\sqrt{t}$ where $z$ is the standard normal variate (or safety factor) corresponding to the required service level (one minus the probability that a stock out will occur) in the time interval $t \in (0, 30)$.

**Fill-Rate Safety Stock** When considering the fill rate, one seeks to determine a product’s expected excess demand, which cannot be filled from inventory so that (1) below holds, where the LHS, $f$, is the required fill rate for the item under consideration.

\[ f = \frac{q}{q + e} \quad (1) \]

The quantity on hand, $q$, at the beginning of the inventory cycle is defined in (2). The right hand side of (2) is the sum of the mean demand between production runs and the safety stock required to meet the required fill rate, $f$.

\[ q = dt + z\sigma\sqrt{t} \quad (2) \]

The expected shortage, $e$, is the demand expected to exceed $q$ given $z\sigma\sqrt{t}$ units of safety stock, is defined in (3). The denominator in (1) is the expected demand
given $q$ units were in stock.

$$e = E(z)\sigma\sqrt{t}$$  \hspace{1cm} (3)

where, $E(z)$, given in (4), is the partial expectation evaluated at a value of $z$ that satisfies the identity in (1).

$$E(z) = \frac{1}{2\pi} \int_{z}^{\infty} (x - z) e^{-\frac{x^2}{2}} dx$$  \hspace{1cm} (4)

We combine (1), (2), (3), and (4) to yield (5), below.

$$E(z) - \left( \frac{1 - f}{f} \right) \left( \frac{dt}{\sigma\sqrt{t}} + z \right) = 0$$  \hspace{1cm} (5)

We can now express $t$ in terms of $z$ as follows in (6) below

$$\sqrt{t} = \left[ \left( \frac{f}{1-f} \right) E(z) - z \right] \left( \frac{\sigma}{d} \right)$$  \hspace{1cm} (6)

**Property 1.** An increase (decrease) in $t$ requires a decrease (increase) in $z$ to maintain the identity in (1).

*Proof.* Using (1) and (3), we can write (7) below.

$$\frac{dt + z\sigma\sqrt{t}}{dt + z\sigma\sqrt{t} + E(z)\sigma\sqrt{t}} = f$$  \hspace{1cm} (7)

Because $\frac{dE(z)}{dz} < 0$ [2], maintaining the identity in (7) requires a reduction in the value of $z$ to raise the value of the denominator of the LHS of ?? when $t$, in the numerator, increases.

**Property 2.** The safety stock function, $ss = z\sigma\sqrt{t}$, increases with $t$, reaches a maximum point and then decreases.

*Proof.* By Property 1 $z$ is a decreasing function of $t$ for any given fill-rate, $ss$
continues to rise until \( z = 0 \) where \( t_0 = \left[ \frac{\sigma}{\sqrt{2\pi}} \left( \frac{f}{1-f} \right) \right]^2 \frac{1}{2\pi} \). With \( z < 0 \) to the right of \( t_0 \), ss is decreasing \( \forall t > t_0 \).

Property 1 can be seen on the 95%, 97%, and 98% fill-rate curves in Figure 1. The hollow circles on these curves are points where the value of \( z \) is zero, that is, where the identity in (8) holds. Safety stock requirements drop below zero as shown on the 95% and 97% fill-rate curves to the right of the filled circles, where the order quantity, \( q \), required to satisfy \( f \) is smaller than the mean demand the replenishment cycle.

\[
\sqrt{T} = \frac{1}{\sqrt{2\pi}} \left( \frac{\sigma}{d} \right) \left( \frac{f}{1-f} \right)
\]

(8)

**Fill Rate Safety Factors.** Property 3 of the fill-rate safety factor will be useful in the development of an efficient solution algorithm.

**Property 3.** (The Fill-Rate Safety Factor Upper Bound)
\[
\forall f \in (0,1), \exists z_f \ni \left( \frac{f}{1-f} \right) E \left( z_0^f \right) - z_0^f = 0 \text{ (see (7)).}
\]

Proof. When \( z = z_0^f \), the value of \( t \) is 0. Values of \( z \geq z_0^f \) imply \( \sqrt{T} \leq 0 \) and are therefore not in the domain of values of \( z \) for \( f \).

Property 3 is illustrated in Figure 2. Safety factor curves for \( f = 90\% \) (dark gray), 95\% (gray), and 98\% (black) displayed for an item with \( d = 1000 \) and \( \sigma = 400 \). When \( t = 0 \), \( z_{90\%}^0 = 0.901, z_{95\%}^0 = 1.159, \) and \( z_{98\%}^0 = 1.485 \).
Figure 1: Requirements of (a) service level safety stock (dashed curves), and (b) fill-rate safety stock (solid curves) as a function of time for an item with $d = 1000$ units per period, and $\sigma = 400$ units per period.
Figure 2: Upper bound of the safety factor, $z$, determined exclusively by the fill rate, $f$. 

\[ \sqrt{t} \]

\[ f=98\% \]
\[ f=95\% \]
\[ f=90\% \]
2 The Family Planning Problem with Fill Rates (FPP-FR)

The following parameters are inputs to the FPP-FR:

- $S_i$: Setup time for family $i$, $i \in \mathbb{N}$
- $s_{ij}$: Setup time for the $j^{th}$ item in family $i$, $(i,j) \in \mathbb{N} \times \mathbb{N}$
- $A_i$: Setup cost for family $i$
- $a_{ij}$: Setup cost for item $j$ in family $i$
- $d_{ij}$: Demand mean for item $j$ in family $i$
- $\sigma_{ij}$: Demand standard deviation for item $j$ in family $i$
- $p_{ij}$: Production rate for item $j$ in family $i$
- $\rho_{ij} = \frac{d_{ij}}{p_{ij}}$: Production capacity needed by item $j$ in family $i$
- $\rho = \sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} \frac{d_{ij}}{p_{ij}}$: Total production capacity required
- $h_{ij}$: Inventory carry cost for item $j$ in family $i$
- $f_{ij}$: Required fill rate for item $j$ in family $i$
- $z_{ij}$: Standard deviation from $\mathcal{N}(0,1)$ corresponding to $f_{ij}$

The decision variables of the problem are:

- $T$: Length of basic period
- $K_i$: Multiplier of family $i$
- $k_{ij}$: Multiplier for item $j$ in family $i$

The following notation will appear in the discussion:

- $C(\cdot)$: Cost function evaluated at $\cdot$
- $G(\cdot)$: Feasibility constraint (see the formulation below)
- $F(\cdot)$: Fill rate identity constraint (see the formulation below)
- $\mathbb{N}$: Set of subscripts of $i$ or $j$ from 1 to $N$
- $\mathbf{K}$: $N$-vector of $K_i$s
$k$ \quad $N \times N$–matrix of $k_{ij}$s

$k_i$. Vector of all item multipliers in family $i$

$\mathbb{P} = \{2^p : p \in \mathbb{Z}_+\}$, set of integer powers of two.

$T_{ij} = TK_i k_{ij}$

**Problem Formulation**  The firm’s objective is to find $(T, K, k)$ so as to minimize the total average cost, given in (9).

$$C(T, K, k) = \sum_{i \in N} \left( \frac{A_i}{TK_i k_{ij}} + \sum_{j \in N} \frac{a_{ij}}{TK_i k_{ij}} + b_{ij} TK_i k_{ij} + g_{ij} \sqrt{TK_i k_{ij}} \right)$$  \quad (9)

$$G(T, K, k) = \sum_{i \in N} S_i K_i + \sum_{i \in N} \sum_{j \in N} s_{ij} K_i k_{ij} - (1 - \rho) T \leq 0$$  \quad (10)

$$F(f_{ij}, d_{ij}, \sigma_{ij}, T_{ij}) = \sqrt{T_{ij}} - \left( \frac{\sigma_{ij}}{d_{ij}} \right) \left[ E(z_{ij}) \left( \frac{f_{ij}}{1 - f_{ij}} \right) - z_{ij} \right] = 0$$  \quad (11)

**Problem M**

The objective of Problem M is to find $(T, K, k, z)$ so as to

Minimize \quad (9)

Subject to \quad (10)

(11); \quad \forall (i, j) \in N \times N

$K_i \in \mathbb{P}; \quad \forall j \in \mathbb{N}$  \quad (12)

$K_{ij} \in \mathbb{P}; \quad \forall (i, j) \in N \times N$  \quad (13)

$T \geq 0$  \quad (14)
where,

\[ b_{ij} = \frac{1}{2} h_{ij} d_{ij} (1 - \rho) \]  

(15)

\[ g_{ij} = \begin{cases} 
  h_{ij} z_{ij} \sigma_{ij} & \text{if } z_{ij} \geq 0 \\
  \frac{1}{2} h_{ij} z_{ij} \sigma_{ij} & \text{otherwise} 
\end{cases} \]  

(16)

Constraint (10) is a necessary feasibility condition. To ensure that the safety stock quantities are set to levels that satisfy the required fill rate for each item, we require that the identity in (11) be enforced by making it a constraint. The Constraints in (12) and (13) require that the family and item multipliers be integer-powers-of-two (IPOT). While a strictly positive value of \( T \) is sought, the strict inequality can be written as a weak one as shown in (14) without loss of accuracy.

The continuous relaxation and solution properties are analogous to [1]. We now proceed to the solution procedure.

### 3 The Multiple Family Algorithm

The steps for solving Problem F are

1. solve Problem S \( n^2 \) times; for each item \((i, j) \in \mathbb{N} \times \mathbb{N}\) run
   - Algorithm 1, and
   - Algorithm 2

2. employ Algorithm 3 \( n^2 \) times; for each family \( i \in \mathbb{N} \)
   - find the KKT points of the continuous relaxation of Problem F
   - select the KKT point, \( m^* \), with the lowest objective value, \( c_i^m(x_i^*, y^*) \).
   - \( \sum_{i \in \mathbb{N}} c_i^m(x_i^*, y^*) \) is a lower bound for Problem F.

3. employing Algorithm 4, execute the multi-family roundoff algorithm to
   - compute \( n^2 \) trial values of \((T, K, k)\)
   - select the cost minimizing \((T^*, K^*, k^*)\)
For a tolerance of $\epsilon = 0.0001$, Algorithm 1 usually requires five to seven iterations to provide an answer. For a tolerance of $l = 0.0001$, Algorithm 2 will solve problem S in $n$ iterations, where $n$ is the smallest integer satisfying the inequality in (17) [3].

\[(0.618)^{n-1} \leq \frac{l}{b-a}\]  

(17)

For fill-rates of 92% and higher, $n = 24$. Below 92%, $n = 23$. The partial expectation in (4) can be expressed as follows [2]:

\[E(z) = \phi(z) - zF(z)\]  

(18)

where,

- $\phi(z)$ Standard normal p.d.f. evaluated at $z$
- $\Phi(z)$ Area under the standard normal curve to the left of $z$
- $F(z)$ Area under the standard normal curve to the right of $z$; that is $F(z) = 1 - \Phi(z)$

The Newton-Raphson Method employs $f(z)$ and $f'(z)$ using (19) and (20).

\[f(z) = \left[ \left( \frac{f}{1-f} \right) E(z) - z \right] \left( \frac{\sigma}{d} \right) - \sqrt{l}\]  

(19)

\[f'(z) = \left[ - \left( \frac{f}{1-f} \right) F(z) - 1 \right] \left( \frac{\sigma}{d} \right)\]  

(20)
Algorithm 1 Procedure to find $z^f_t$ using the Newton-Raphson Method

Require:
$f, d, \sigma$ //item properties, subscripts supressed
$t$ //to find $z^f_t$ set $t = 0$

1: Initialize
   $\epsilon \leftarrow 0.0001$
   $z_0 \leftarrow 3$

   Begin Newton-Raphson Method
2: $z_1 \leftarrow z_0 - \frac{f(z_0)}{f'(z_0)}$
3: while $|z_1 - z_0| > \epsilon$ do
4:   $z_0 \leftarrow z_1$
5:   $z_1 \leftarrow z_0 - \frac{f(z_0)}{f'(z_0)}$
6: end while
7: $z^f_t \leftarrow z_1$
8: return $z^f_t$

End
Algorithm 2 Solution Procedure for the Single Item Problem

Require:
\[ \beta \leftarrow \frac{\sqrt{5} - 1}{2} \quad \alpha \leftarrow 1 - \beta \quad \varepsilon \leftarrow 0.0001 \quad (z^0_a, z^0_b) \leftarrow (-3, z^b) \]

Equations
1: \[ z^* \leftarrow \frac{(z_k^a + z_k^b)}{2} \]
2: \[ t^* \leftarrow (\frac{\sigma^2}{2}) \left[ E(z^*) \left( \frac{r}{1 - r} \right) - z^* \right]^2 \]

BEGIN
3: \( (z^1_a, z^1_b) = (z^0_a, z^0_b) \)
4: for \( k = 1 \) to 24 do
5: \[ \delta^k \leftarrow z^k_b - z^k_a \]
\[ (\lambda^k, \mu^k) \leftarrow (z^k_b + \alpha \delta^k, z^k_b + \beta \delta^k) \]
\[ \delta^k \leftarrow \mu^k - \lambda^k \]
6: if \( \delta^k \leq \varepsilon \) then
7: return \( z^* \) and \( t^* \) //Using the equations in statements 1 & 2
Stop: EXIT algorithm
8: else
9: \[ t^k_\lambda \leftarrow (\frac{\sigma^2}{2}) \left[ E(\lambda^k) \left( \frac{r}{1 - r} \right) - \lambda^k \right]^2 \]
\[ c(\lambda^k) \leftarrow \frac{\sigma^2}{\lambda^k} + \frac{1}{\lambda^k} h d t^k_\lambda + g^k_\lambda \sqrt{t^k_\lambda} \]
\[ c(\mu^k) \leftarrow \frac{\sigma^2}{\mu^k} + \frac{1}{\mu^k} h d t^k_\mu + g^k_\mu \sqrt{t^k_\mu} \]
//use (16) to compute \( g^k_\lambda \) //use (16) to compute \( g^k_\mu \)
10: \( (z^k_a, z^k_b) \leftarrow \begin{cases} (\lambda^k, z^k_b) & \text{if } c(\lambda^k) > c(\mu^k) \\ (z^k_a, \mu^k) & \text{otherwise} \end{cases} \)
11: end if
12: end for
13: return \( z^* \) and \( t^* \) //Using the equations in statement 1 & 2
Stop: EXIT algorithm
Algorithm 3 Find and Test KKT Points

1: Initialize
   \( \forall j \in \mathbb{N}, y_j \leftarrow t_j \)
   reindex \( y \) so that \( y_1 \leq y_2 \leq y_n \)
   set \( M = \{ m : \text{the solution produced in iteration } m \text{ is feasible} \} = \emptyset \)

2: Functions

3: \( c_m (t, y) = \bar{a}_m t + \bar{b}_m t + \sum_{j=m+1}^{n} \left( \frac{a_j}{y_j} + b_j y_j \right) + \sum_{j \in \mathcal{M}} (g_j \sqrt{y_j}) \)

4: for \( m = 1 \) to \( n \) do

5: \( \bar{a}_m = A + a_1 + \cdots + a_m \)

6: \( \bar{b}_m = b_1 + \cdots + b_m \)

7: end for

8: Compute \((\check{t}_1, \check{z}_1)\) \(//\) Using Algorithm 1 and Algorithm 2

9: if \( \check{t}_1 \geq y_2 \) then

10:   Constraint (11) in [1] is not satisfied; infeasible solution, not a KKT point.

11: else

12: \( t \leftarrow \bar{y}_1 \leftarrow \check{t}_1 \)

13: \( \mathcal{M} \leftarrow \{ 1 \} \)

14: \( \mathcal{C} \leftarrow c_1 (t, y) \) \(//\) use the function in statement 3

15: end if

16: for \( m = 2 \) to \( n \) do

17:   Create dummy item \( m_0 \) by duplicating item \( m \) and making one change:

18:   \( a_{m_0} = \bar{a}_m \)

19: Compute \((\check{t}_m, \check{z}_m)\) for the dummy item \(//\) Using Algorithm 1 and Algorithm 2

20: Compute \((\check{t}_m, \check{z}_m, \check{z}_m)\) \(//\) Using Algorithm 2 and Algorithm 5

21: if \( \check{t}_m \leq y_{m+1} \) then

22:   for \( j = 1 \) to \( m \) do

23:     \( t \leftarrow \bar{y}_j \leftarrow \check{t}_j \)

24:   end for

25: \( \mathcal{M} \leftarrow \{ m \} \)

26: \( \mathcal{C} \leftarrow c_m (t, y) \) \(//\) use the function in statement 3

27: //When \( t_m = y_{m+1} \), Constraint (11) in [1] is not satisfied; infeasible solution, not a KKT point.

28: end if

29: end for

30: return \( m^* = \arg \min_{m \in \mathcal{M}} \mathcal{C} = \{ c_m (t, y) : (t, y) \text{ is a KKT point} \} \)

Stop: EXIT algorithm
Algorithm 4 Multi-Family Roundoff Algorithm

1: Data
   \((t^*, y^*)\)
2: Initialize
   \(\gamma \leftarrow t^*\) //Begin Roundoff Procedure
3: for \((i, j) \in \mathbb{N} \times \mathbb{N}\) do
4:   Find \(t^{ij}\) and \(\pi^{ij} \in \mathbb{Z}_+ \ni y_{ij} = t^{ij} 2^{\pi^{ij}}\) and \(t^{ij} \in [\gamma, 2\gamma)\)
5:   end for
6: Add a subscript \(h\) to \(t^{ij}\) and \(\pi^{ij} \ni t^{ij}_h \leq t^{ij}_{h+1}\)
7: for \(\phi = 1\) to \(n^2\) do
8:   \(\pi^{ij}_\phi \leftarrow \begin{cases} \pi^{ij}_h - 1 & \text{if } h \leq \phi \\ \pi^{ij}_h & \text{otherwise} \end{cases}\)
9:   \(k^{ij}_\phi \leftarrow 2^{\pi^{ij}_\phi}\)
10: for \(j = 1\) to \(n\) do
11:   Compute \(z^{ij}_\phi\) using Algorithm 1
12: end for
13: \(t^*_\phi = \arg\min_t c\{t^*_\phi, k^*_\phi\}\)
14: end for
Algorithm 5 Golden Section Search along $t$

Require:
\[
(t_{a_0}, t_{b_0}) \leftarrow (t_{m}, y_1) \quad \delta^0 = t_{b_0} - t_{a_0} \quad \varepsilon \leftarrow 0.0001
\]
\[
\omega = 1 \quad \beta \leftarrow \frac{\sqrt{5} - 1}{2} \quad \alpha \leftarrow 1 - \beta
\]

Compute $\omega$, the required number of iterations

1: while $0.618^{(\omega - 1)} > \frac{\varepsilon}{\sqrt{5}}$ do
2: \quad $\omega \leftarrow \omega + 1$
3: end while
4: \quad $(t_{a_1}, t_{b_1}) \leftarrow (t_{a_0}, t_{b_0})$
5: for $k = 1$ to $\omega$ do
6: \quad $\delta^k \leftarrow t_{b_k} - t_{a_k}$
7: \quad $(\lambda^k, \mu^k) \leftarrow (t_{a_k} + \delta^k \alpha, t_{b_k} + \delta^k \beta)$
8: \quad $\delta^k \leftarrow \mu^k - \lambda^k$
9: \quad if $\delta^k \leq \varepsilon$ then
10: \quad \quad $(t_{a_{k+1}}, t_{b_{k+1}}) \leftarrow (\lambda^k, \mu^k)$
11: \quad else
12: \quad \quad $c_{\lambda_k} \leftarrow \overline{a} \overline{m} \lambda$
13: \quad \quad $c_{\mu_k} \leftarrow \overline{a} \overline{m} + \overline{b} \overline{m} \mu$
14: \quad \quad for $j = 1$ to $m$ do
15: \quad \quad \quad compute $z_{\lambda_k}^j$ using Algorithm 1
16: \quad \quad \quad \quad $c_{\lambda_k} \leftarrow c_{\lambda_k} + g_j \sqrt{\lambda^k}$
17: \quad \quad \quad compute $z_{\mu_k}^j$ using Algorithm 1
18: \quad \quad \quad \quad $c_{\mu_k} \leftarrow c_{\mu_k} + g_j \sqrt{\mu^k}$
19: \quad \quad \quad end for
20: \quad \quad $(t_{a_{k+1}}, t_{b_{k+1}}) \leftarrow \begin{cases} (\lambda^k, z_{\lambda_k}^k) & \text{if } c_{\lambda_k}^k > c_{\mu_k}^k \\ (z_{a_k}^k, \mu^k) & \text{otherwise} \end{cases}$
21: \quad end if
22: end for
23: $t^*_m = (t_{a_k} + t_{b_k+1})/2$
24: $c^*_m = \overline{a} \overline{m} + \overline{m} t^*_m$
25: for $j = 1$ to $m$ do
26: \quad compute $z_{t_m}^j$ using Algorithm 1
27: \quad $c^*_m \leftarrow c^*_m + g_j \sqrt{t_m}$
28: end for
29: return $t^*_m$ and $c^*_m$
4 Example

Algorithm F is illustrated with an example. In this example, a facility produces five families, each with five items. The time unit was arbitrarily chosen to be one week. The data were generated from the uniformly distributed parameters in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family setup time (weeks)</td>
<td>$U(0.015, 0.025)$</td>
</tr>
<tr>
<td>Family setup cost ($)</td>
<td>$U(500, 1000)$</td>
</tr>
<tr>
<td>Item setup time (weeks)</td>
<td>$U(0.0042, 0.0125)$</td>
</tr>
<tr>
<td>Item setup cost ($)</td>
<td>$U(100, 500)$</td>
</tr>
<tr>
<td>Item holding cost ($)</td>
<td>$U(0.10, 1.25)$</td>
</tr>
<tr>
<td>Item demand mean (units)</td>
<td>$U(1000, 2500)$</td>
</tr>
<tr>
<td>Item demand standard deviation (% of demand mean)</td>
<td>$U(0.60, 0.90)$</td>
</tr>
<tr>
<td>Item production rate (units/week)</td>
<td>$U(50000, 200000)$</td>
</tr>
<tr>
<td>Item fill rate (%)</td>
<td>$U(0.95, 0.9999)$</td>
</tr>
</tbody>
</table>

Example 1.

\[
S = \begin{bmatrix}
0.0243 & 0.0249 & 0.0210 & 0.0236 & 0.0182 \\
\end{bmatrix}
\]

\[
A = \begin{bmatrix}
967 & 643 & 634 & 712 & 559 \\
\end{bmatrix}
\]

\[
s = \begin{bmatrix}
0.0110 & 0.0047 & 0.0106 & 0.0044 & 0.0121 \\
0.0120 & 0.0086 & 0.0060 & 0.0118 & 0.0125 \\
0.0109 & 0.0123 & 0.0078 & 0.0118 & 0.0118 \\
0.0095 & 0.0056 & 0.0104 & 0.0117 & 0.0114 \\
0.0118 & 0.0113 & 0.0104 & 0.0081 & 0.0045
\end{bmatrix}
\]
\[ h = \begin{bmatrix} $286 & $389 & $398 & $477 & $163 \\ $368 & $292 & $229 & $328 & $302 \\ $330 & $475 & $499 & $340 & $489 \\ $180 & $181 & $296 & $204 & $294 \\ $194 & $306 & $396 & $171 & $379 \end{bmatrix} \]

\[ a = \begin{bmatrix} $0.17 & $0.25 & $0.39 & $1.09 & $1.22 \\ $0.48 & $0.54 & $0.67 & $0.51 & $0.22 \\ $1.19 & $0.46 & $0.25 & $1.22 & $0.15 \\ $1.03 & $0.16 & $1.06 & $0.52 & $0.20 \\ $0.71 & $1.25 & $0.54 & $0.50 & $0.53 \end{bmatrix} \]

\[ d = \begin{bmatrix} 1,084 & 1,629 & 1,999 & 1,248 & 1,267 \\ 1,140 & 1,708 & 2,190 & 2,342 & 2,383 \\ 1,471 & 2,180 & 1,541 & 1,251 & 1,362 \\ 1,156 & 1,049 & 1,141 & 1,711 & 2,317 \\ 1,655 & 2,392 & 2,213 & 1,378 & 1,689 \end{bmatrix} \]

\[ \sigma = \begin{bmatrix} 912.94 & 1,128.57 & 1,384.11 & 1,057.80 & 1,021.33 \\ 746.81 & 1,430.45 & 1,869.60 & 1,796.31 & 2,143.03 \\ 915.11 & 1,613.20 & 1,273.69 & 870.07 & 895.79 \\ 845.27 & 695.17 & 1,017.89 & 1,402.85 & 1,762.08 \\ 1,091.80 & 1,783.18 & 1,791.56 & 954.95 & 1,023.16 \end{bmatrix} \]

\[ p = \begin{bmatrix} 77,992 & 193,190 & 96,706 & 148,305 & 136,860 \\ 85,259 & 156,193 & 116,702 & 57,720 & 146,190 \\ 62,443 & 179,797 & 118,258 & 92,120 & 192,021 \\ 168,194 & 166,679 & 141,044 & 73,565 & 59,973 \\ 72,447 & 187,138 & 64,699 & 142,463 & 60,164 \end{bmatrix} \]
Computation results are shown using two procedures (a) We first use algorithm F as given in §3, next (b) we apply the multi-family algorithm to the corresponding deterministic problem with demand means used as demand rates. Once the solution is obtained, we add safety stocks and compute their costs.

Table 2: Computation results of the Fill Rate Algorithm

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>(1)*</th>
<th>(2)**</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total average cost</td>
<td>TC=46, 904.71</td>
<td>49, 481.40</td>
</tr>
<tr>
<td>% Difference</td>
<td>5.49%</td>
<td></td>
</tr>
<tr>
<td>Total average Family setup cost</td>
<td>$6, 435.15</td>
<td>$3, 771.36</td>
</tr>
<tr>
<td>Total average Item setup cost</td>
<td>$11, 003.05</td>
<td>$7, 863.53</td>
</tr>
<tr>
<td>Total average working stock holding cost</td>
<td>$8, 038.05</td>
<td>$11, 634.89</td>
</tr>
<tr>
<td>Total average safety stock holding cost</td>
<td>$21, 428.46</td>
<td>$26, 211.62</td>
</tr>
<tr>
<td>Basic period length</td>
<td>$T^* = 0.323$</td>
<td>0.932</td>
</tr>
<tr>
<td>Family multipliers</td>
<td>$K^* = [2 \ 1 \ 2 \ 2 \ 2 \ 1 \ 1 \ 1 \ 1 \ 1]$</td>
<td></td>
</tr>
<tr>
<td>Item multipliers</td>
<td>$k^* = [1 \ 2 \ 1 \ 1 \ 1 \ 2 \ 1 \ 1 \ 1 \ 1 \ 2 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 2 \ 1 \ 1]$</td>
<td></td>
</tr>
</tbody>
</table>

*(1) Multi-family algorithm  
**(2) Deterministic multi-family with safety stocks computed at the end

Using algorithm F, which considers safety stock costs explicitly, resulted in cost savings of 5.49% over algorithm (2). This represents annual savings of $133,988.33 over algorithm (2). The average cost savings over 50 randomly
generated problems with the distribution of parameters given above were 4.27% over algorithm (2). This represents annual dollar savings of $139,075.87. The solution performance was, on average, 0.9997% above the lower bound, with a standard deviation of 0.5571% and a maximum of 2.2565%.

Algorithm F generated solutions whose basic period length was, on average, 62.16% smaller than that of algorithm (2), consistent with the trends in practice whereby reductions in working stock and safety stock levels and their associated costs drive lot sizes to be smaller and production to be more frequent.

5 Summary

In this technical note, we have extended the Multiple Family Economic Lot Scheduling Problem to explicitly include the consideration of safety stocks for a fill rate criterion. We exploited a number of properties of the problem to develop an efficient solution procedure. The form of our solution is a basic period cyclic schedule, with item multipliers restricted to integer-powers-of-two. We solved a representative set of sample problems using our procedure, and compared the results to a solution procedure that ignored the safety stock costs. Our results exhibited total costs typically 4.27% lower than the alternative procedure.

There are a number of directions for future research on the MFELSP-SS for fill rates. There is compelling reason to believe that, unlike the multi-family algorithm—which solves the MFELSP-SS for service level, algorithm F may very well produce solutions that favor longer basic periods over algorithm (2). This is because the safety factor is a decreasing function of the cycle time. Consequently, safety stock levels begin to decrease with increasing cycle time. Computational studies to investigate this and the impact of other parameters on the algorithm’s performance would be useful.

Another useful study would be a computational study that compares the
performance of the algorithm for the service level safety stocks and the one for fill rates.

References

