Generating a Sales Forecast With a Simple Depth-of-Repeat Model

Peter S. Fader
www.petefader.com

Bruce G. S. Hardie[†]
www.brucehardie.com

May 2004

1. Introduction

Central to diagnosing the performance of a new product is the decomposition of its total sales into trial, first repeat, second repeat, and so on, components:

$$S(t) = T(t) + R_1(t) + R_2(t) + R_3(t) + \cdots$$

where $S(t)$ is the cumulative sales volume up to time t (assuming that only one unit is purchased on each purchase occasion), $T(t)$ equals the cumulative number of people who have made a trial purchase by time t, and $R_j(t)$ denotes the number of people who have made at least j repeat purchases of the new product by time t $(j = 1, 2, 3, \ldots)$.

- We can decompose $T(t)$ in the following manner:

$$T(t) = NF_0(t) \tag{1}$$

  where N is number of customers whose purchases are being monitored and $F_0(t)$ is the proportion of customers who have made their trial purchase by t.

- We can decompose the $R_j(t)$ by conditioning on the time at with the $(j-1)$th purchase occurred:[1]

$$R_j(t) = \sum_{t_{j-1}=j}^{t-1} F_j(t \,|\, t_{j-1})\big[R_{j-1}(t_{j-1}) - R_{j-1}(t_{j-1} - 1)\big] \tag{2}$$

[1] Eskin, Gerald J. (1973), "Dynamic Forecasts of New Product Demand Using a Depth of Repeat Model," *Journal of Marketing Research*, **10** (May), 115–129.

where $F_j(t \mid t_{j-1})$ is the proportion of customers who have made a jth repeat purchase by t, given that their $(j-1)$th repeat purchase was made in period t_{j-1}, and $R_{j-1}(t_{j-1}) - R_{j-1}(t_{j-1}-1)$ is the number of individuals who made their $(j-1)$th repeat purchase in time period t_{j-1}. (Note that $R_0(t) = T(t)$ and $R_j(t) = 0$ for $t \leq j$.)

Equations (1) and (2) are simply definitional. If we specify mathematical expressions for $F_0(t)$ and the $F_j(t \mid t_{j-1})$, we arrive at a model of new product sales. In our 2004 ART Forum tutorial "Forecasting Repeat Buying for New Products and Services", we present the following model, with separate submodels for trial, first repeat (denoted by $FR(t)$ instead of $R_1(t)$) and additional repeat ($AR(t) = R_2(t) + R_3(t) + \cdots$):

For trial, we have

$$T(t) = NP(\text{trial by } t) \tag{3}$$

$$P(\text{trial by } t) = p_0(1 - e^{-\theta_T t}) \tag{4}$$

For first repeat, we have

$$FR(t) = \sum_{t_0=1}^{t-1} P(\text{first repeat by } t \mid \text{trial at } t_0)\big[T(t_0) - T(t_0 - 1)\big] \tag{5}$$

$$P(\text{first repeat by } t \mid \text{trial at } t_0) = p_1\big(1 - e^{-\theta_{FR}(t-t_0)}\big) \tag{6}$$

For additional repeat $(j \geq 2)$, we have

$$AR(t) = \sum_{j=2}^{\infty} R_j(t) \tag{7}$$

$$\begin{aligned} R_j(t) = \sum_{t_{j-1}=j}^{t-1} \Big\{ & P(j\text{th repeat by } t \mid (j-1)\text{th repeat at } t_{j-1}) \\ & \times \big[R_{j-1}(t_{j-1}) - R_{j-1}(t_{j-1}-1)\big]\Big\} \end{aligned} \tag{8}$$

$$\begin{aligned} & P(j\text{th repeat by } t \mid (j-1)\text{th repeat at } t_{j-1}) \\ & \qquad = p_j\big(1 - e^{-\theta_{AR}(t-t_{j-1})}\big) \end{aligned} \tag{9}$$

$$p_j = p_\infty(1 - e^{-\gamma j}) \tag{10}$$

As demonstrated in the tutorial, it is very easy to calibrate the model parameters within a spreadsheet environment. For the Kiwi Bubbles dataset (from a panel of $N = 1499$ households) with a 24-week calibration period, we have

p_0	θ_T	p_1	θ_{FR}	p_∞	γ	θ_{AR}
0.08620	0.06428	0.36346	0.46140	0.78158	1.00140	0.23094

Given these parameter estimates, we can first generate a forecast of trial, then a forecast of first repeat (conditional on the trial forecast), then a forecast of second repeat (conditional on the first-repeat forecast), and so on. For a forecast horizon of t_f periods, we should (in theory) allow for up to $t_f - 1$ levels of repeat[2] and the process of "coding up" (3)–(10) in a spreadsheet environment is rather cumbersome. It is *much* easier to write a small stand-alone program to do the task.

In this note, we present programs written in Perl and MATLAB for generating the sales forecast.[3] Any interested reader should have no difficultly in "translating" one of these programs into his/her favourite language, be it Basic, C, Fortran, SAS/IML, etc.

2. Forecasting Trial Transactions

Our goal is to generate a (cumulative) sales forecast for the new product up to the end of the year (week 52). We start by generating a forecast of $T(t)$, the cumulative number of trial transactions by t, for $t = 1, \ldots, 52$.

Given $\hat{p}_0$ and $\hat{\theta}_T$, we substitute (2) and (1) and loop over t. In Perl, we using the following code:

```
my $N = 1499;    # number of panelists
my $endwk = 52;  # length of forecast period
my $p_0 = 0.08620;
my $theta_T = 0.06428;
for ($t = 1; $t <= $endwk; $t++){
  $trial[$t] = $N*$p_0*(1-exp(-$theta_T*$t));
}
```

Given that the basic data element in MATLAB is an array, there is no need to loop over time; we simply specify the vector of time points at which we want to evaluate (1) and (2):

```
N = 1499;     % number of panelists
endwk = 52;   % length of forecast period
p_0 = 0.08620;
theta_T = 0.06428;
trial = N*p_0*(1-exp(-theta_T*[1:endwk]'));
```

[2]Under the assumption that a customer can have only one transaction per unit of time, the earliest point in time a first repeat purchase can occur is period 2. Following this logic, the summation limit in (7) should really be $t - 1$, not ∞.

[3]There is nothing magical about our choice of these programming environments. Why Perl? It's free and is ideally suited for "quick and dirty" programming tasks such as this one. Furthermore, it is very easy to pick up if you have any programming experience. Why MATLAB? It's a very powerful modelling environment that we use in our own research, development, and analysis activities.

3. Forecasting First-Repeat Transactions

The next step is to generate a forecast of first-repeat purchasing, conditional on our forecast of trial transactions. We have generated a forecast of $T(t)$, the cumulative number of trial transactions by t, whereas our expression for $FR(t)$, (5), requires the incremental number of triers in each period. We compute this quantity, storing it in the array `eligible`. At the time time, we use (6) to compute the probability of making a first repeat purchase t periods after a trial purchase ($t = 1, \ldots, 52$). Given these two quantities, we use (5) to compute $FR(t)$, looping over t. In Perl, we using the following code:

```
my $p_1 = 0.36346;
my $theta_FR = 0.46140;
for ($t = 1; $t <= $endwk; $t++){
  if ($t gt 1){
    $eligible[$t]=$trial[$t]-$trial[$t-1];
  } else {
    $eligible[$t]=$trial[$t];
  }
  $prob[$t] = $p_1*(1-exp(-$theta_FR*$t));
}
for ($t = 1; $t <= $endwk; $t++){
  $repeat[$t] = 0;
  for ($k = 1; $k <= $t-1; $k++){
    $repeat[$t]=$repeat[$t]+$eligible[$k]*$prob[$t-$k];
  }
}
```

The array-based nature of MATLAB means we are able to replace the double loop over $\$t$ and $\$k$ with a single multiplication operation:

```
p_1 = 0.36346;
theta_FR = 0.46140;
eligible = [trial(1);diff(trial)];
prob = p_1*(1-exp(-theta_FR*[1:endwk]'));
for i = 1:endwk
    probmat(:,i) = [zeros(i,1); prob(1:endwk-i)];
end;
repeat(:,1) = probmat*eligible;
```

4. Forecasting Additional Repeat Transactions

To forecast additional repeat, we first compute second repeat (conditional on our forecast of first repeat), then compute third repeat (conditional on our forecast of second repeat), and so on. The code we use to accomplish this is basically the same as that for first repeat, embedded within a loop over depth-of-repeat levels. Given the assumption that a customer can have only one transaction per period, we could theoretically have up to 51 depth-of-repeat levels for a 52-week forecast horizon. Another consequence of this assumption is that the first week in which a jth repeat purchase could occur is week $j+1$; this means $R_j(t) = 0$ for $t \leq j$.

We compute (8)–(10) for each of the depth-of-repeat levels using the following Perl code:

```
my $p_inf = 0.78158;
my $gamma = 1.00140;
my $theta_AR = 0.23094;
for ($dor = 2; $dor <= $endwk-1; $dor++){
  $p_j = $p_inf*(1-exp(-$gamma*$dor));
  for ($t = 1; $t <= $endwk; $t++){
    if ($t gt 1){
      $eligible[$t] = $repeat[($dor-2)*$endwk+$t]
        -$repeat[($dor-2)*$endwk+$t-1];
    } else {
      $eligible[$t] = $repeat[($dor-2)*$endwk+$t];
    }
    $prob[$t] = $p_j*(1-exp(-$theta_AR*$t));
  }
  for ($t = 1; $t <= $endwk; $t++){
    $repeat[($dor-1)*$endwk+$t] = 0;
    for ($k = $dor; $k <= $t-1; $k++){
      $repeat[($dor-1)*$endwk+$t]
        = $repeat[($dor-1)*$endwk+$t]+$eligible[$k]*$prob[$t-$k];
    }
  }
}
```

In MATLAB, these calculations are performed using the following code:

```
p_inf = 0.78158;
gamma = 1.00140;
theta_AR = 0.23094;
p_j = p_inf*(1-exp(-gamma*[2:endwk-1]));
for dor = 2:endwk-1
    eligible = [repeat(1,dor-1);diff(repeat(:,dor-1))];
    prob = p_j(dor-1)*(1-exp(-theta_AR*[1:endwk]'));
    for i = 1:endwk
       probmat(:,i) = [zeros(i,1); prob(1:endwk-i)];
    end;
    repeat(:,dor) = probmat*eligible;
end;
```

Note that the MATLAB code stores the repeat-sales forecasts in a two-dimensional array (week $\times$ depth-of-repeat level). Perl has no support for multidimensional arrays. While can we can emulate a two-dimensional array by creating an array that contains references to other arrays, our Perl code takes the less elegant but more transparent approach of creating a one-dimensional array with $51 \times 52 = 2652$ elements. The first 52 elements contain the first-repeat numbers for weeks $1, \ldots, 52$; the second 52 elements contain the second-repeat numbers for weeks $1, \ldots, 52$; and so on.

5. Bringing It All Together

We can now compute $AR(t) = R_2(t) + \cdots + R_{52}(t)$, and print out our forecasts of $T(t)$, $FR(t)$, $AR(t)$, and $S(t)$ $(= T(t) + FR(t) + AR(t))$. Our complete Perl code is given in Figure 1; the corresponding MATLAB code is given in Figure 2. The forecasts generated by these programs are presented in Figures 3 and 4, respectively. As would be expected, these forecasts are identical.

```
#!perl
# forecast.pl -- generates a sales forecast using a
# simple depth-of-repeat model

use strict;
my ($t, $k, $dor, $p_j);
my (@trial, @repeat, @ar, @totsls, @eligible, @prob);

my $N = 1499;    # number of panelists
my $endwk = 52;  # length of forecast period

# Generate forecast of expected trial sales
# trial model parameters
my $p_0 = 0.08620;
my $theta_T = 0.06428;
for ($t = 1; $t <= $endwk; $t++){
  $trial[$t] = $N*$p_0*(1-exp(-$theta_T*$t));
}

# Generate forecast of expected first repeat sales
# FR model parameters
my $p_1 = 0.36346;
my $theta_FR = 0.46140;
for ($t = 1; $t <= $endwk; $t++){
  if ($t gt 1){
    $eligible[$t]=$trial[$t]-$trial[$t-1];
  } else {
    $eligible[$t]=$trial[$t];
  }
  $prob[$t] = $p_1*(1-exp(-$theta_FR*$t));
}
for ($t = 1; $t <= $endwk; $t++){
  $repeat[$t] = 0;
  for ($k = 1; $k <= $t-1; $k++){
    $repeat[$t]=$repeat[$t]+$eligible[$k]*$prob[$t-$k];
  }
}

# Generate forecast of additional repeat sales
# AR model parameters
my $p_inf = 0.78158;
my $gamma = 1.00140;
my $theta_AR = 0.23094;
for ($dor = 2; $dor <= $endwk-1; $dor++){
  $p_j = $p_inf*(1-exp(-$gamma*$dor));
  for ($t = 1; $t <= $endwk; $t++){
    if ($t gt 1){
      $eligible[$t] = $repeat[($dor-2)*$endwk+$t]-$repeat[($dor-2)*$endwk+$t-1];
    } else {
      $eligible[$t] = $repeat[($dor-2)*$endwk+$t];
    }
    $prob[$t] = $p_j*(1-exp(-$theta_AR*$t));
  }
  for ($t = 1; $t <= $endwk; $t++){
    $repeat[($dor-1)*$endwk+$t] = 0;
    for ($k = $dor; $k <= $t-1; $k++){
      $repeat[($dor-1)*$endwk+$t] = $repeat[($dor-1)*$endwk+$t]+$eligible[$k]*$prob[$t-$k];
    }
  }
}

# Compute total additional repeat and total sales and
# display results
for ($t = 1; $t <= $endwk; $t++){
  $ar[$t] = 0;
  for ($dor = 2; $dor <= $endwk-1; $dor++){
    $ar[$t] = $ar[$t] + $repeat[($dor-1)*$endwk+$t];
  }
  $totsls[$t] = $trial[$t] + $repeat[$t] + $ar[$t];
}
for ($t = 1; $t <= $endwk; $t++){
  printf "%3d  %12.4f %12.4f  %12.4f %12.4f\n", $t, $trial[$t], $repeat[$t], $ar[$t], $totsls[$t];
}
```

Figure 1: Complete Perl code (`forecast.pl`)

```
% forecast.m -- generates a sales forecast using a
% simple depth-of-repeat model

N = 1499;      % number of panelists
endwk = 52;    % length of forecast period

% Generate forecast of expected trial sales
% trial model parameters
p_0 = 0.08620;
theta_T = 0.06428;
trial = N*p_0*(1-exp(-theta_T*[1:endwk]'));

repeat = zeros(endwk,endwk-1);
probmat = zeros(endwk,endwk);

% Generate forecast of expected first repeat sales
% FR model parameters
p_1 = 0.36346;
theta_FR = 0.46140;
eligible = [trial(1);diff(trial)];
prob = p_1*(1-exp(-theta_FR*[1:endwk]'));
for i = 1:endwk
    probmat(:,i) = [zeros(i,1); prob(1:endwk-i)];
end;
repeat(:,1) = probmat*eligible;

% Generate forecast of additional repeat sales
% AR model parameters
p_inf = 0.78158;
gamma = 1.00140;
theta_AR = 0.23094;
p_j = p_inf*(1-exp(-gamma*[2:endwk-1]));
for dor = 2:endwk-1
    eligible = [repeat(1,dor-1);diff(repeat(:,dor-1))];
    prob = p_j(dor-1)*(1-exp(-theta_AR*[1:endwk]'));
    for i = 1:endwk
       probmat(:,i) = [zeros(i,1); prob(1:endwk-i)];
    end;
    repeat(:,dor) = probmat*eligible;
end;

% Compute total additional repeat and total sales and
% display results
ar = sum(repeat(:,2:endwk-1),2);
totsls = trial + sum(repeat,2);
disp([[1:endwk]' trial repeat(:,1) ar totsls]);
```

Figure 2: Complete MATLAB code (`forecast.m`)

```
D:\>perl -w forecast.pl
  1       8.0445        0.0000         0.0000        8.0445
  2      15.5882        1.0807         0.0000       16.6689
  3      22.6623        2.7753         0.1507       25.5883
  4      29.2959        4.7939         0.5296       34.6194
  5      35.5166        6.9575         1.1700       43.6441
  6      41.3500        9.1571         2.0785       52.5856
  7      46.8201       11.3274         3.2463       61.3938
  8      51.9498       13.4304         4.6562       70.0363
  9      56.7601       15.4452         6.2866       78.4918
 10      61.2708       17.3615         8.1144       86.7467
 11      65.5008       19.1755        10.1161       94.7924
 12      69.4674       20.8873        12.2693      102.6240
 13      73.1871       22.4992        14.5525      110.2388
 14      76.6752       24.0151        16.9457      117.6359
 15      79.9461       25.4393        19.4306      124.8159
 16      83.0134       26.7765        21.9903      131.7802
 17      85.8897       28.0315        24.6096      138.5308
 18      88.5870       29.2090        27.2745      145.0705
 19      91.1163       30.3137        29.9726      151.4025
 20      93.4882       31.3498        32.6923      157.5303
 21      95.7124       32.3216        35.4237      163.4577
 22      97.7981       33.2331        38.1576      169.1888
 23      99.7539       34.0878        40.8859      174.7277
 24     101.5880       34.8894        43.6013      180.0787
 25     103.3080       35.6411        46.2974      185.2465
 26     104.9208       36.3460        48.9686      190.2354
 27     106.4332       37.0070        51.6099      195.0501
 28     107.8515       37.6269        54.2168      199.6952
 29     109.1815       38.2082        56.7856      204.1752
 30     110.4286       38.7533        59.3129      208.4949
 31     111.5981       39.2645        61.7961      212.6587
 32     112.6948       39.7439        64.2326      216.6713
 33     113.7233       40.1934        66.6205      220.5371
 34     114.6877       40.6149        68.9581      224.2606
 35     115.5920       41.0102        71.2440      227.8463
 36     116.4401       41.3809        73.4774      231.2983
 37     117.2354       41.7284        75.6573      234.6211
 38     117.9811       42.0544        77.7832      237.8187
 39     118.6804       42.3601        79.8547      240.8952
 40     119.3362       42.6467        81.8718      243.8548
 41     119.9512       42.9155        83.8345      246.7012
 42     120.5278       43.1675        85.7429      249.4383
 43     121.0686       43.4039        87.5973      252.0698
 44     121.5757       43.6255        89.3982      254.5995
 45     122.0512       43.8334        91.1461      257.0308
 46     122.4972       44.0283        92.8417      259.3671
 47     122.9153       44.2111        94.4856      261.6120
 48     123.3074       44.3825        96.0786      263.7685
 49     123.6752       44.5432        97.6216      265.8400
 50     124.0200       44.6939        99.1155      267.8294
 51     124.3433       44.8352       100.5612      269.7398
 52     124.6466       44.9678       101.9597      271.5740

D:\>
```

Figure 3: Output generated by forecast.pl

```
>> forecast
    1.0000    8.0445         0         0    8.0445
    2.0000   15.5882    1.0807         0   16.6689
    3.0000   22.6623    2.7753    0.1507   25.5883
    4.0000   29.2959    4.7939    0.5296   34.6194
    5.0000   35.5166    6.9575    1.1700   43.6441
    6.0000   41.3500    9.1571    2.0785   52.5856
    7.0000   46.8201   11.3274    3.2463   61.3938
    8.0000   51.9498   13.4304    4.6562   70.0363
    9.0000   56.7601   15.4452    6.2866   78.4918
   10.0000   61.2708   17.3615    8.1144   86.7467
   11.0000   65.5008   19.1755   10.1161   94.7924
   12.0000   69.4674   20.8873   12.2693  102.6240
   13.0000   73.1871   22.4992   14.5525  110.2388
   14.0000   76.6752   24.0151   16.9457  117.6359
   15.0000   79.9461   25.4393   19.4306  124.8159
   16.0000   83.0134   26.7765   21.9903  131.7802
   17.0000   85.8897   28.0315   24.6096  138.5308
   18.0000   88.5870   29.2090   27.2745  145.0705
   19.0000   91.1163   30.3137   29.9726  151.4025
   20.0000   93.4882   31.3498   32.6923  157.5303
   21.0000   95.7124   32.3216   35.4237  163.4577
   22.0000   97.7981   33.2331   38.1576  169.1888
   23.0000   99.7539   34.0878   40.8859  174.7277
   24.0000  101.5880   34.8894   43.6013  180.0787
   25.0000  103.3080   35.6411   46.2974  185.2465
   26.0000  104.9208   36.3460   48.9686  190.2354
   27.0000  106.4332   37.0070   51.6099  195.0501
   28.0000  107.8515   37.6269   54.2168  199.6952
   29.0000  109.1815   38.2082   56.7856  204.1752
   30.0000  110.4286   38.7533   59.3129  208.4949
   31.0000  111.5981   39.2645   61.7961  212.6587
   32.0000  112.6948   39.7439   64.2326  216.6713
   33.0000  113.7233   40.1934   66.6205  220.5371
   34.0000  114.6877   40.6149   68.9581  224.2606
   35.0000  115.5920   41.0102   71.2440  227.8463
   36.0000  116.4401   41.3809   73.4774  231.2983
   37.0000  117.2354   41.7284   75.6573  234.6211
   38.0000  117.9811   42.0544   77.7832  237.8187
   39.0000  118.6804   42.3601   79.8547  240.8952
   40.0000  119.3362   42.6467   81.8718  243.8548
   41.0000  119.9512   42.9155   83.8345  246.7012
   42.0000  120.5278   43.1675   85.7429  249.4383
   43.0000  121.0686   43.4039   87.5973  252.0698
   44.0000  121.5757   43.6255   89.3982  254.5995
   45.0000  122.0512   43.8334   91.1461  257.0308
   46.0000  122.4972   44.0283   92.8417  259.3671
   47.0000  122.9153   44.2111   94.4856  261.6120
   48.0000  123.3074   44.3825   96.0786  263.7685
   49.0000  123.6752   44.5432   97.6216  265.8400
   50.0000  124.0200   44.6939   99.1155  267.8294
   51.0000  124.3433   44.8352  100.5612  269.7398
   52.0000  124.6466   44.9678  101.9597  271.5740

>>
```

Figure 4: Output generated by forecast.m