Supplements to

Optimal Sales Schemes against Interdependent Buyers

Masaki Aoyagi
ISER, Osaka University

1 Property of the Optimal Sequential Scheme

Under the optimal scheme identified in Theorem 2, the response by each buyer moves in the direction of the response dominant among their predecessors. In other words, a buyer accepts with a higher probability than his predecessor when most buyers before him have chosen to accept, and rejects with a higher probability when most buyers before him have chosen to reject. Formally, the following proposition states that when the state $\alpha_{t-1} > 0$ as a result of many buyers having accepted, the expected probability that the next buyer accepts is higher than the probability that the current buyer accepts. Conversely, when the state $\alpha_{t-1} < 0$ as a result of many buyers having rejected, the expected probability that the next buyer rejects is higher than the probability that the current buyer rejects.

Proposition S.1. *Suppose that every s_i has the uniform distribution, that (5) and (6) hold, and that $\underline{s} + c_0 = 0$. Then the following hold under the optimal sequential scheme σ. For any $\delta > 0$, there exists $\bar{\varepsilon} > 0$ such that if $\varepsilon < \bar{\varepsilon}$, then $\alpha_{t-1} > \delta$ implies $E^\sigma[z_{t+1}(\tilde{\alpha}_t) \mid \alpha_{t-1}] > z_t(\alpha_{t-1})$, and $\alpha_{t-1} < -\delta$ implies $E^\sigma[z_{t+1}(\tilde{\alpha}_t) \mid \alpha_{t-1}] < z_t(\alpha_{t-1})$.*

Proof of Proposition S.1 Since z_t is an affine function by Theorem A.4, we have (A.29) for any α, and hence the expected value of the probability z_{t+1} of acceptance in period $t+1$ conditional on the state α_{t-1} at the beginning of period t equals

$$E^\sigma[z_{t+1}(\tilde{\alpha}_t) \mid \alpha_{t-1}] = z_{t+1}(\alpha_{t-1}).$$

It hence follows from $a_t = \Delta$ and (A.33) that

$$\begin{aligned} E^\sigma[z_{t+1}(\tilde{\alpha}_t) \mid \alpha_{t-1}] - z_t(\alpha_{t-1}) &= \frac{1}{2\Delta}\left\{a_{t+1} - a_t + (b_{t+1} - b_t)\,\alpha_{t-1}\right\} \\ &= \frac{1}{2\Delta}\,(c_{t+1} - c_t)\,\alpha_{t-1} + o(\varepsilon). \end{aligned}$$

Since $c_{t+1} - c_t > 0$, we obtain the desired result.

Buyer order				Revenue
1	2	3	4	1.0168
2	1	3	4	1.0161
3	1	2	4	1.0124
4	1	2	3	1.0082

Table 1: Uniform Distribution

2 Optimal Sequential Schemes in Other Environments: Numerical Examples

In this section, we report results of numerical computation which find that the conclusion of Theorem 2 on the optimal ordering of buyers extends to more general environments. Set the number of buyers $I = 4$ and their dependence weights $c_i = i/7$ for $i = 1, \ldots, 4$. Our strategy is to use Lemma A.2 to compare the revenue associated with every contingent scheme. That is, we begin with the sequential pricing problems for every pair of buyers and every initial state $\alpha_0 \in C_{I-2} = C_2$, then proceed to the problems with three buyers, and so on.[25]

In the first example, the private signal s_i has the uniform distribution over the unit interval, but c_i's do not satisfy (4). We find that the optimal sequential scheme (among all schemes and not just the one with fixed orders) trades with buyers $1, \ldots, 4$ in this order. Table 1 lists the seller's revenue from the four alternative buyer sequences in (A.18).

In the second example, we assume that a buyer's private signal s_i has a truncated exponential distribution over $[0, M]$ $(M > 0)$. Specifically, we suppose that the cumulative distribution F and the associated density f are respectively given by

$$F(s_i) = \frac{1 - e^{-\delta s_i}}{1 - e^{-\delta M}} \quad \text{and} \quad f(s_i) = \frac{\delta e^{-\delta s_i}}{1 - e^{-\delta M}} \quad \text{for } s_i \in [0, M].$$

For $M = 2$ and $\delta = 0.5$, we again find that the optimal sequential scheme trades in the increasing order of the dependence weights. Table 2 lists the seller's revenue from the four buyer sequences in (A.18).

Listed below is the matlab program to compute the seller's revenue along the four alternative sequences in (A.18) for the truncated exponential distribution reported

[25] Revenue is computed for each point on the grid of the size $1/200 \times 1/200$ over the (α_{t-1}, z_t) space for each $t = 1, \ldots, 4$. See below for a program source in Matlab for the representative cases.

Buyer order				Revenue
1	2	3	4	1.5892
2	1	3	4	1.5873
3	1	2	4	1.5799
4	1	2	3	1.5633

Table 2: Truncated Exponential Distribution

in Table 2. The subroutine follows the main program.

```
%%%%%%%%%%%%%%%% Main program %%%%%%%%%%%%%%%%

close all; clear all; diary([mfilename '.out']);
disp('optimal sales scheme model. F: truncated exponential distribution');

%Solve the sequential pricing problems with J=I and alpha_0=0.

denom = 7;
%permutations of four buyers
B1=[1/denom 2/denom 3/denom 4/denom];
B2=[2/denom 1/denom 3/denom 4/denom];
B3=[3/denom 1/denom 2/denom 4/denom];
B4=[4/denom 1/denom 2/denom 3/denom];


% parameters
M = 2; %least upper bound of the distribution
delta = 0.5;
edm = exp(-delta*M);
mu = (delta^(-1) - (M+delta^(-1))*edm) / (1-edm);

% define grid for alpha
maxa = 3*(M-mu); % maximum value of alpha grid
mina = 3*(-mu); % minimum value of alpha grid
ia = 0.005;  % size of alpha grid
na = round((maxa - mina)/ia+1); % number of alpha grid points

% determine grid point for alpha=0
for i=1:na
if i*ia + mina <= 0 & (i+1)*ia + mina >= 0;
a0=i+1;
end
end


% define vec_a as a column vector
for i=1:na
vec_a(i,1) = (i-1)*ia + mina;
end

%%%
B=B1;
A = aoyagi11func(B, maxa, mina, ia, na, M, delta, edm, mu);
A1=A;
disp(A1(1,a0));

%B=B2
B=B2;
A = aoyagi11func(B, maxa, mina, ia, na, M, delta, edm, mu);
```

```
A2=A;
disp(A2(1,a0));

%B=B3
B=B3;
A = aoyagi11func(B, maxa, mina, ia, na, M, delta, edm, mu);
A3=A;
disp(A3(1,a0));

%B=B4
B=B4;
A = aoyagi11func(B, maxa, mina, ia, na, M, delta, edm, mu);
A4=A;
disp(A4(1,a0));


%evaluate at the initial state a=0

disp('A1-A2');
disp(A1(1,a0)-A2(1,a0));

disp('A1-A3');
disp(A1(1,a0)-A3(1,a0));

disp('A1-A4');
disp(A1(1,a0)-A4(1,a0));

diary off;

%%%%%%%%%%%%%%%% Subroutine %%%%%%%%%%%%%%%%

function A = aoyagi11func(B, maxa, mina, ia, na, M, delta, edm, mu)

nb = 4; %number of buyers

% define grid for z
maxz = 1;  % maximum value of z grid
minz = 0;  % minimum value of z grid
iz = 0.005;  % size of z grid
nz = round((maxz - minz)/iz+1); % number of z grid points

% define z as a column vector
for i=1:nz
z(i,1) = (i-1)*iz + minz;
end

% set parameter values
c0 = 0;
ci = ones(nb,1);
for i = 1 : nb;
ci(i,1) = B(1,i);
end;

%initialize payoff vectors
pi = ones(nz,na);  % values when (z,a)
pistar = ones(nb,na);  % optimal value when state is a
pistara = ones(nz,1); % expected optimal value when accepted
pistarr = ones(nz,1); % expected optimal value when rejected

%t=4 start from the final period
t=4;
maxat = (t-1)*(M-mu); % maximum of alpha in t
minat = (t-1)*(-mu); % minimum of alpha in t
for i = 1:na
a = mina + (i-1)*ia; % compute alpha for the ith grid point

for j = 1:nz;
```

```
theta = -1/delta * log(1-(1-edm)*(1-z(j,1))); % theta for each z
edth = exp(-delta*theta);

if a > maxat | a < minat % if the ith grid point exceeds the bounds
pi(:,i) = -100*nb; % substitute a small value
else
pi(j,i) = z(j,1)*theta + ci(t)*a*z(j,1) + c0*z(j,1);
end

end;

end

pistar(t,:) = max(pi);

%t=1,2,3
t=3;
while t>0
maxat = (t-1)*(M-mu);
minat = (t-1)*(-mu);
for i = 1:na
a = mina + (i-1)*ia;

k = ones(nz,1); % kappa(z) as a nz vector
l = ones(nz,1); % lambda(z) as a nz vector

for j = 1:nz;

theta = -1/delta * log(1-(1-edm)*(1-z(j,1))); % theta for each z
edth = exp(-delta*theta);

if z(j,1) == 0
k(j,1) = M-mu;
else
k(j,1) = ((theta + delta^(-1))*edth-(M+delta^(-1))*edm) / (edth-edm)...
- (delta^(-1)-(M+delta^(-1))*edm) / (1-edm);
end;

if z(j,1) == 1
l(j,1) = -mu;
else
l(j,1) = (delta^(-1)-(theta+delta^(-1))*edth) / (1-edth)...
- (delta^(-1)-(M+delta^(-1))*edm) / (1-edm);
end;

aa(j) = a+k(j,1); % future state for each z when accepted
ar(j) = a+l(j,1); % future stete for each z when refected

if aa(j) > t*(M-mu);
aa(j) = t*(M-mu);
end;
if ar(j) < t*(-mu);
ar(j) = t*(-mu);
end;

gaa = fix((aa(j) - mina)/ia) + 1; % grid point for aa(j)
gar = fix((ar(j) - mina)/ia) + 1; % grid point for ar(j)

pistara(j,1) = pistar(t+1,gaa); % future payoff pistar_a when accepted
pistarr(j,1) = pistar(t+1,gar); % future payoff pistar_r when rejected

if a > (maxat+ia) | a < (minat-ia)
pi(j,i) = -100*nb;
else
pi(j,i) = z(j,1)*theta + (ci(t)*a+c0)*z(j,1) + ...
z(j,1)*pistara(j,1) + (1-z(j,1))*pistarr(j,1);
end
```

```
end
end

pistar(t,:) = max(pi); % choose the maximizing z for each a in period t
t=t-1;

end
A(1,:) = pistar(1,:);
```