Experimental evolution of anticipatory regulation in *E. coli*.
E. coli utilizes the transporter RhaT to internalize rhamnose (Baldoma et al., 1990a; Garciamartin et al., 1992). Thereafter, the sugar is processed by metabolic enzymes RhaB, RhaA, and RhaD (Moralejo et al., 1993). The regulon is controlled by two AraC-like transcription factors, RhaR and RhaS (Holcroft and Egan, 2000; Wickstrum and Egan, 2004; Wickstrum et al., 2007). Both bind upstream of the coding sequence of the genes essential for rhamnose utilization. The fact that rhamnose utilization regulon has two regulators and that the coding regions of one of the transcription factors (RhaR) and the transporter (RhaT) overlap make the rhamnose utilization regulon unique in E. coli (Baldoma et al., 1990b; Via et al., 1996; Wickstrum and Egan, 2004). The rhamnose-utilization regulon is also under catabolite repression, when glucose is present in the environment (Egan and Schleif, 1993).

Prokaryotes protect themselves against oxidative stress ($\text{H}_2\text{O}_2$, OH free radical, $\text{O}_2^-$) with the action of superoxide dismutases, catalases, and peroxiredoxin (Lushchak, 2001). In the presence of oxidative stress, the expression of these enzymes is up-regulated by transcription factors, OxyR (Zheng et al., 1998), SoxS, and SoxR (Nunoshiba et al., 1992). In a recent study, more than 50 distinct binding sites were reported for OxyR, SoxS and SoxR, when E. coli was studied under paraquat stress (Seo et al., 2015). Most of these binding sites were found to be RpoD (Sigma 70)-dependent, and combined, the three transcription factors controlled more than 100 genes in response to oxidative stress. The OxyR, SoxRS regulon includes activation of zwf (which encodes glucose-6 phosphate-1-dehydrogenase), increase in cellular NADPH pool (Henard et al., 2010; Seo et al., 2015), aromatic amino acid production, and upregulation in cell wall biosynthesis (Pomposiello and Demple, 2002).
S2. Range of values of parameters used in the simulation.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b$ (for all promoters)</td>
<td>0 to 0.3</td>
</tr>
<tr>
<td>$k_{on}$ (for all DNA-protein interactions)</td>
<td>0 to 60</td>
</tr>
<tr>
<td>$k_{off}$ (for all DNA-protein interactions)</td>
<td>0.3</td>
</tr>
<tr>
<td>$kd$ (degradation rate constant)</td>
<td>0 to 0.015</td>
</tr>
<tr>
<td>Signal (S1 or S2)</td>
<td>0 or 1</td>
</tr>
<tr>
<td>$b_{max}$ (maximum benefit)</td>
<td>1.5</td>
</tr>
<tr>
<td>$Km$ (half-maximum benefit)</td>
<td>15</td>
</tr>
<tr>
<td>$co$ (cost of synthesis of one protein molecule)</td>
<td>0.01</td>
</tr>
</tbody>
</table>
S3. Prior exposure of Alt lines to rhamnose leads to a faster soxS induction.

Figure. In the presence of paraquat, lines Alt1 (A), Alt2 (B), and Alt3 (C) exhibited a faster induction of the soxS promoter when brought from M9 glycerol media containing rhamnose (red) compared to cells brought from M9 glycerol media (blue). The rhamnose-dependent faster induction of the soxS promoter is absent in the paraquat-evolved lines (PQ1 (D), PQ2 (E), and PQ3 (F)) and in rhamnose-evolved lines (Rha1 (G), Rha2 (H), and Rha3 (I)). All experiments were done in triplicate. The average of the three experiments and the standard deviation is reported. (J) Ancestor cells, when transitioned to M9 glycerol with paraquat from M9 glycerol media with (red) or without rhamnose (blue) do not exhibit any statistically significant difference in the induction kinetics from the soxS promoter. (* indicates p-value < 0.01, Two-tailed, paired, t-test)
S4. List of mutations in the three Rhamnose-PQ alternating lines.

<table>
<thead>
<tr>
<th>Line Alt1</th>
<th></th>
<th>Mutation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. potA</td>
<td></td>
<td>Frameshift mutation (c.764dupA p.Asn255fs)</td>
</tr>
<tr>
<td>2. trkH</td>
<td></td>
<td>Missense mutation. (251G&gt;A. Gly84Asp)</td>
</tr>
<tr>
<td>3. glpK</td>
<td></td>
<td>Missense mutation. (175T&gt;C. Ser59Pro)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line Alt2</th>
<th></th>
<th>Mutation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. rpoS</td>
<td></td>
<td>Missense mutation (377G&gt;A. Gly126Glu)</td>
</tr>
<tr>
<td>2. ygeH</td>
<td></td>
<td>Missense mutation. (713A&gt;C. Glu238Ala)</td>
</tr>
<tr>
<td>3. glpK</td>
<td></td>
<td>Missense mutation. (184G&gt;T. Val62Leu)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line Alt3</th>
<th></th>
<th>Mutation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. cyaA</td>
<td></td>
<td>Frameshift mutation (c.1436_1442delATCAGCC p.His479fs)</td>
</tr>
<tr>
<td>2. rrsA</td>
<td></td>
<td>Noncoding transcript variant</td>
</tr>
<tr>
<td>3. glpK</td>
<td></td>
<td>Missense mutation. (973A&gt;C. Asn325His)</td>
</tr>
</tbody>
</table>
S5. Codes for simulations.

```matlab
for i=1:1:1
maxb = 6; kntp = 15; konal = random('Uniform',60,60); koffa = random('Uniform',0.3,0.3); a(i,1) = 0 + (i-1)*0.001; c0 = 0.01; [fit,bs,ks] = Random(maxb,kntp,c0,a(i),konal,koffa); fitness(i,1) = fit(1); fitness(i,2) = fit(2); b(i,1) = bs(1); b(i,2) = bs(2); b(i,3) = bs(3); k(i,1) = ks(1); k(i,2) = ks(2); k(i,3) = ks(3); k(i,4) = ks(4); con(i,1) = 0; if(fitness(i,1)<fitness(i,2)) con(i) = 1; end end
% plot(a,con,'o'); [X,Y] = meshgrid(0:0.005:0.3,0:0.005:0.3); for i = 1:1:61 for j=1:1:61 x = X(i,j); y = Y(i,j); Z(i,j) = Simple_Signal_Target2(maxb,kntp,c0,a,x,y,60,0.3,60,0.3,0.01,1,1,konal,koffa)-fit(2); end % end surf(X,Y,Z); xlabel('b1'); ylabel('b2'); zlabel('δF'); %title(''); parameters = 'test.mat'; save(parameters) direction = [1 0 0]; rotate(hsurf,direction,25)
```

```
function [r,bs,kc] = Random(maxb,kntp,c0,a,konal,koffa)
maxf = zeros(2,1); rtotal = 0.01; s1 = 1; s = 1; bs = zeros(3,1); kc = zeros(4,1); for kondna = 60:1:60 for koffdna = 0.3:0.005:0.3 for bn = 0:0.005:0.3 fitness = Simple_Signal_Target(maxb,kntp,c0,a,bn,konal,koffdna,rtotal,s,kona,koffa); if(fitness>maxf(1)) maxf(1) = fitness; bs(1) = bn; end end end end
```

```
c = 1; for kondna = 60:1:60 for koffdna = 0.3:0.1:0.3 for kondna1 = 60:1:60
```

for b1 = 0:0.005:0.3
for bc = 0:0.005:0.3
    fitness = Simple_Signal_Target2(maxb,kmtp,c0,a,b1,bc,kondna,koffdna,rtotal,s1,s,kona,koffa1,kona,ko ffa);
    \% hkl2fit(c,:) = [b1 bc fitness];
    \% c = c+1;
    if(fitness>maxf(2))
        maxf(2) = fitness;
        bs(2) = b1;
        bs(3) = bc;
        kc = [kondna;koffdna;kondna;koffdna];
    end
end
end
end
end
end

%plot(a,bs(2)','*');
%hold on;
%plot(a,bs(3)','.);
%xlabel('degradation rate a');
%ylabel('Transcription rate b');
%r = maxf;
%legend('b1','b2');


########################################################################## Function file → No conditioning case
##########################################################################

function answer2 = Simple_Signal_Target(maxb,kmtp,c0,a,b,kondna,koffdna,rtotal,s,kona,koffa)
\%Target Protein 2
n = 1;               \%no. of signal molecules binding to activator protein
s = 0.005;           \%signal concentration
kona = 100;          \%k_on of signal-activator complex
koffa = 0.2;           \%k_off of signal-activator complex
kxa = koffa/kona;    \%ratio
rtotal = 10;         \%total concentration of activator protein
\%main parameters
b = 1.5;               \%maximal translation rate
a = 0.08;              \%degradation rate of target protein
maxb = 4;             \%maximal benefit conferred by target protein
kmtp = 10;             \%concentration at which half-maximal benefit is conferred
%0 = 0.5;             \%cost of production for one cell
a1 = 1;               \%no. of proteins produced by cell per unit time

t = zeros(200,1);       \%time
tspan = zeros(200,1);
for i=1:1:200
    t(i) = i;
    tspan(i) = i;
end
ract_steady = ((s^n)*rtotal)/(kxa + (s^n)); %concentration of activated protein at steady state
ract = ract_steady*(1-exp(-koffa*(kxa+s)*x))*(1-prob);
\%solve the ode, to get tp, as a function of t
p = polyfit(x,prob,3);
[x,tp] = odes23s(@(x,tp) b*(p(1)*x^3 + p(2)*x^2 + p(3)*x + p(4)) - a*tp, tspan,0);
%plot(t,ben,'r');
%hold on;
%plot(t,cost,'b');
%overall cost
r = maxf;

for i=1:1:4
    tf = ((i-1)*t(200)/4); %dividing into five segments
    ti = 1 + (((i-1)*t(200)/4)); %lower limit
   惩戒(i) = ti;
   惩戒(5) = tf;
   惩戒(3) = (ti+tf)/2;
end
\[ tc(2) = \frac{(ti+tc(3))}{2}; \]
\[ tc(4) = \frac{(tc(3)+tf)}{2}; \]
\[ ropl = zeros(5,1); \] % amount of protein at different times
for j=1:1:5
\[ tc(j) = round(tc(j)); \]
\[ ropl(j) = prob(tc(j))*b; \] % calculating rate of production at each point
end
\[ tp1 = \frac{(tf-ti)/6}{*ropl(1)+4*ropl(3)+ropl(5)}; \]
\[ tp2 = \frac{(tf-ti)/12}{*ropl(1)+4*ropl(2)+2*ropl(3)+4*ropl(4)+ropl(5)}; \]
\[ pt = \frac{16*tp2}{15} - tp1/15; \] % total in this segment
\[ tp\_total = tp\_total + pt; \] % summing up all five segments
end
\[ ocost = tp\_total*c0/a1; \] % cost after signal stops
extrat = 1;
\[ [x,prob] = ode23s(@(x,prob) kondna*ract(200)*exp(-koffa*x)*(1-prob) - koffdna*prob, tspan, pv); \]
\[ tspan = 0; \]
for j=1:1:200
  rop_f = prob(j)*b;
  t(200+j) = 200+j;
  if(rop_f<0.001)
    extrat = j;
    for k=1:1:j
      tspane(k) = k;
    end
  break
  end
end
if(size(tspane)<2)
  for i=1:1:200
    tspane(i) = i;
  end
end
\[ [x,prob] = ode23s(@(x,prob) kondna*ract(200)*exp(-koffa*x)*(1-prob) - koffdna*prob, tspan, pv); \]
\[ p = polyfit(x,prob,3); \]
\[ y = polyval(p,x); \]
\[ [x,prob] = ode23s(@(x,prob) kondna*ract(200)*exp(-koffa*x)*(1-prob) - koffdna*prob, tspan, tp(200)); \]
\[ for j=1:1:extrat
  tp(200+j) = tpe(j);
end
end
\[ tf = extrat; \]
\[ ti = 1; \]
\[ tc(1) = ti; \]
\[ tc(5) = tf; \]
\[ tc(3) = \frac{(ti+tf)}{2}; \]
\[ tc(2) = \frac{(ti+tc(3))}{2}; \]
\[ tc(4) = \frac{(tc(3)+tc(5))}{2}; \]
\[ rope = zeros(5,1); \] % target protein concentration after signal stops
for i=1:1:5
\[ tc(i) = round(tc(i)); \]
\[ rope(i) = prob(tc(i))*b; \] % rate of production after signal stops at all 5 values
end
\[ tp1 = \frac{(tf-ti)/6}{*rop(1)+4*rop(3)+rop(5)}; \]
\[ tp2 = \frac{(tf-ti)/12}{*rop(1)+4*rop(2)+2*rop(3)+4*rop(4)+rop(5)}; \]
\[ tp\_ex = \frac{16*tp2}{15} - tp1/15; \] % total target protein produced after signal stopping
\[ coste = tp\_ex*c0/a1; \] % extra cost after signal stops
\[ ocost = ocost + coste; \] % overall benefit
\[ oben = 0; \]
for i=1:1:4
\[ tf = i*t(200)/4; \] % final point
\[ ti = 1 + ((i-1)*t(200)/4); \] % initial point
\[ tc(1) = ti; \]
\[ tc(5) = tf; \]
\[ tc(3) = \frac{(ti+tf)}{2}; \]
\[ tc(2) = \frac{(ti+tc(3))}{2}; \]
\[ tc(4) = \frac{(tc(3)+tc(5))}{2}; \]
\[ pben = zeros(5,1); \] % pseudo benefit, to store benefit of each segment
for j=1:1:5
\[ tc(j) = round(tc(j)); \]
\[ pben(j) = \frac{maxb*tp(tc(j))}{(kmtp + tp(tc(j)))}; \]
end
\[ pben1 = \frac{(tf-ti)/6}{*pben(1)+4*pben(3)+pben(5)}; \]
\[ pben2 = \frac{(tf-ti)/12}{*pben(1)+4*pben(2)+2*pben(3)+4*pben(4)+pben(5)}; \]
\[ sben = \frac{16*pben2}{15} - pben1/15; \] % benefit conferred in this segment
\[ oben = oben + sben; \] % summing up all benefits conferred at all times
end
\[ oben = oben/200; \]
\[ answer2 = oben - ocost; \]
function answer = Simple_Signal_Target2(maxb, kmtp, c0, a, b1, b, kondna1, koffdna1, kondna, koffdna, rtotal, s1, s, konal, koffa1, kona, kofa)

n = 1;                %no. of signal molecules binding to activator protein
%total = 10;         %total concentration of activator protein
%s1 = 0.005;          %signal concentration
%maxb = 4;           %maximal benefit conferred by target protein
% kmtp = 15;          %concentration at which half-maximal benefit is conferred
% c0 = 0.5;           %cost of production for one cell
a1 = 1;               %no. of proteins produced by cell per unit time
%kona1 = 60;          %k_on of signal-activator complex
%koffa1 = 0.3;        %k_off of signal-activator complex
kxa = koffa1/kona1;   %ratio
%main parameters
% b = 0.7;            %maximal translation rate
% koffdna1 = 0.3;     %k_off of dna-activator complex
% kondna1 = 10;       %k_on of dna-activator complex
kxdna1 = koffdna1/kondna1;
tspan = zeros(200,1); %time
for i=1:1:200
    t(i) = i;
    tspan(i) = i;
end
ract_steady = ((s1^n)*rtotal)/(kxa + (s1^n));    %conc.
entration of activated protein at steady state
ract = ract_steady*(1-exp(-koffa1*(kxa+s)*t)); %concentration of activated protein
ract1 = ract(200);
[x,prob] = ode23s(@(x,prob) kondna1*ract_steady*(1-exp(-koffa1*(kxa+s)*x))*(1-prob) - koffdna1*prob, tspan, 0);
%solve the ode, to get tp, as a function of t
p = polyfit(x,prob,3);
y = polyval(p,x);
% solve the ode, to get tp, as a function of t
[x,tp] = ode23s(@(x,tp) b1*(p(1)*x^3 + p(2)*x^2 + p(3)*x + p(4)) - a*tp, tspan,0);
% plot(t,ben,'r');
% hold on;
% plot(t,cost,'b');
%overall cost
tp_total = 0;
for i=1:1:4                     %dividing into five segments
    tf = ((i-1)*t(200)/4);  %upper limit of adaptive quadrature
    ti = 1 + ((i-1)*t(200)/4); %lower limit
    tc(1) = ti;
    tc(5) = tf;
    tc(3) = (ti+tf)/2;
    tc(2) = (ti+tc(3))/2;
    tc(4) = (tc(3)+tf)/2;
    ropl(1) = zeros(5,1);   %amount of protein at different times
    for j=1:1:5
        tc(j) = round(tc(j));
        ropl(j) = polyval(p,tc(j));%points at which values are calculated
    end
    t1p = (tf-ti)/6*(ropl(1)+4*ropl(3)+ropl(5));
    t2p = (tf-ti)/12*(ropl(1)+4*ropl(2)+2*ropl(3)+4*ropl(4)+ropl(5));
    ptp = 16*t2p/15 - t1p/15; %total in this segment
    tp_total = tp_total + ptp;   %summing up all five segments
end
ocost = tp_total*c0/a1;
answer1 = -ocost;
end
%%%%%%%%%%%%%%%%% Function file - Conditioning present case

function answer = Simple_Signal_Target2(maxb, kmtp, c0, a, b1, b, kondna1, koffdna1, kondna, koffdna, rtotal, s1, s, konal, koffa1, kona, kofa)

n = 1;                   %no. of signal molecules binding to activator protein
%total = 10;         %total concentration of activator protein
%s1 = 0.005;          %signal concentration
%konal = 100;        %k_on of signal-activator complex
%koffa = 0.2;        %k_off of signal-activator complex
kxa = koffa/konal;   %ratio
%total = 10;         %total concentration of activator protein
%main parameters
% b = 1.5;            %maximal translation rate
% total = 10;         %time

t = zeros(200,1);       %time

for i=1:1:200
    t(i) = i;
    tspan(i) = i;
end
ract_steady = ((s1^n)*rtotal)/(kxa + (s1^n));    %conc.
entration of activated protein at steady state
ract = ract_steady*(1-exp(-koffa1*(kxa+s)*t)); %concentration of activated protein
ract1 = ract(200);
[x,prob] = ode23s(@(x,prob) kondna1*ract_steady*(1-exp(-koffa1*(kxa+s)*x))*(1-prob) - koffdna1*prob, tspan, 0);
%solve the ode, to get tp, as a function of t
p = polyfit(x,prob,3);
y = polyval(p,x);
% solve the ode, to get tp, as a function of t
[x,tp] = ode23s(@(x,tp) b1*(p(1)*x^3 + p(2)*x^2 + p(3)*x + p(4)) - a*tp, tspan,0);
% plot(t,ben,'r');
% hold on;
% plot(t,cost,'b');
%overall cost
tp_total = 0;
for i=1:1:4                     %dividing into five segments
    tf = ((i-1)*t(200)/4);  %upper limit of adaptive quadrature
    ti = 1 + ((i-1)*t(200)/4); %lower limit
    tc(1) = ti;
    tc(5) = tf;
    tc(3) = (ti+tf)/2;
    tc(2) = (ti+tc(3))/2;
    tc(4) = (tc(3)+tf)/2;
    ropl(1) = zeros(5,1);   %amount of protein at different times
    for j=1:1:5
        tc(j) = round(tc(j));
        ropl(j) = polyval(p,tc(j));%points at which values are calculated
    end
    t1p = (tf-ti)/6*(ropl(1)+4*ropl(3)+ropl(5));
    t2p = (tf-ti)/12*(ropl(1)+4*ropl(2)+2*ropl(3)+4*ropl(4)+ropl(5));
    ptp = 16*t2p/15 - t1p/15; %total in this segment
    tp_total = tp_total + ptp;   %summing up all five segments
end
ocost = tp_total*c0/a1;
answer1 = -ocost;
tspan = zeros(200,1);
for i=1:1:200
    t(i) = i;
    tspan(i) = i;
end
ract_f = ract(200);
ract_steady = ((s^n)*rtotal)/(kxa + (s^n)); %concentration of activated protein at steady state
ract = ract_steady*(1 - exp(-koffa*(kxa+s)*t)); %concentration of activated protein

% koffdna = 0.1; % k off of dna-activator complex
% kondna = 100; % k on of dna-activator complex
kxdna = koffdna/kondna;
[x,prob] = ode23s(@(x,prob) kondna*ract_steady*(1-exp(-koffa*(kxa+s)*x))*(1-prob) - koffdna*prob, tspan, 0);
p = polyfit(x,prob,3);
y = polyval(p,x);
pv2 = prob(200);
prob2 = prob;
[x,prob] = ode23s(@(x,prob) kondna1*ract1*(1-exp(-koffa1*x))*(1-prob) - koffdna1*prob, tspan, pv);
p1 = polyfit(x,prob,3);
y1 = polyval(p1,x);
prob1 = prob;

for i=1:1:200
    tp(i+200) = tps2(i);
end
tp_total = 0;
for i=1:1:4
    tf = (i*t(200)/4); % upper limit of adaptive quadrature
ti = 1 + ((i-1)*t(200)/4); % lower limit
tc(1) = ti;
points at which values are calculated
tc(5) = tf;
tc(3) = (ti+tf)/2;
tc(2) = (ti+tc(3))/2;
tc(4) = (tc(3)+tf)/2;
rop1 = zeros(5,1);
rop2 = zeros(5,1);
for j=1:1:5
tc(j) = round(tc(j));
rop1(j) = prob1(tc(j))*b1; % calculating rate of production at each point
rop2(j) = prob2(tc(j))*b;
end
tp1 = ((tf-ti)/6)*((rop1(1)+4*rop1(2)+4*rop1(3)+rop1(5)));
ptp1 = 16*tp2/15 - tp1/15; % total in this segment
end

if(size(tspane)<2)
    for i=1:1:200
        tspane(i) = i;
    end
end
[x,prob] = ode23s(@(x,prob) kondna*ract(200)*exp(-koffa*x)*(1-prob) - koffdna*prob, tspan, pv2);
p = polyfit(x,prob,3);
y = polyval(p,x);
for j=1:1:extrat
    tp(j+200) = tps2(j);
end

% cost after signal stops
extrat = 1;
[x,prob] = ode23s(@(x,prob) kondna*ract(200)*exp(-koffa*x)*(1-prob) - koffdna*prob, tspan, pv2);
tspan = 0;
for j=1:1:200
    rop_f = prob(j)*b;
t(200+j) = 200+j;
    if(rop_f<0.001) % only till concentrations where cost is significant; not required after production ~0
        extrat = j;
        for k=1:1:j
            tspane(k) = k;
        end
        break
    end
end
for i=1:1:200
    tspane(i) = i;
end
[x,prob] = ode23s(@(x,prob) kondna*ract(200)*exp(-koffa*x)*(1-prob) - koffdna*prob, tspan, pv2);
p = polyfit(x,prob,3);
y = polyval(p,x);
for j=1:1:extrat
    tp(j+400) = tps2(j);
end
tf = extrat;
ti = 1;
tc(1) = ti;
tc(5) = tf;
tc(3) = (ti+tf)/2;
tc(2) = (ti+tc(3))/2;
tc(4) = (tc(3)+tc(5))/2;
rope = zeros(5,1);  %target protein concentration after signal stops
for i=1:1:5
  tc(i) = round(tc(i));
  rope(i) = prob(tc(i))*b;  %rate of production after signal stops at all 5 values
end
tpe1 = ((tf-ti)/6)*(rope(1)+4*rope(3)+rope(5));
tpe2 = ((tf-ti)/12)*(rope(1)+4*rope(2)+2*rope(3)+4*rope(4)+rope(5));
tp_ex = 16*tpe2/15 - tpe1/15;  % total target protein produced after signal stopping
ocost = tp_ex*c0/a1;  %extra cost after signal stops
for i=1:1:4  %dividing into 5 segments
  tf = i*t(200)/4;  %final point
  ti = 1 + ((i-1)*t(200)/4);  %initial point
  tc(1) = ti;
tc(5) = tf;
tc(3) = (ti+tf)/2;
tc(2) = (ti+tc(3))/2;
tc(4) = (tc(3)+tf)/2;
pben = zeros(5,1);  %pseudo benefit, to store benefit of each segment
  for j=1:1:5
    tc(j) = round(tc(j));
    pben(j) = maxb*tps2(tc(j))/(kmtp + tps2(tc(j)));
  end
  pben1 = ((tf-ti)/6)*(pben(1)+4*pben(3)+pben(5));
pben2 = ((tf-ti)/12)*(pben(1)+4*pben(2)+2*pben(3)+4*pben(4)+pben(5));
sben = 16*pben2/15 - pben1/15;  %benefit conferred in this segment
  oben = oben + sben;  %summing up all benefits conferred at all times
end
oben = oben/200;
answer2 = oben - ocost;
ocost = ocost - answer1;
for i=1:1:(400+extrat)
t(i)= i;
end
answer = answer1 + answer2;

########################################################################################################## END ##########################################################################################################
S6. Supplement References


