HW7 ECE 719 Optimal systems

Spring 2016
Electrical engineering department
University of Wisconsin, Madison

Instructor: Professor B Ross Barmish

By

Nasser M. Abbasi

December 30, 2019

Contents

List of Figures

List of Tables

0.1 Problem 1

Barmish

ECE 719 – Homework City Planners

A committee of city planners are to recommend the "best" allocation of fire stations to three districts. They will base their decision on *expected property damage* which they hope will be minimal. The table below reflects differences in districts due to factors such as population, socioeconomic makeup, etc. The budget restricts total number of stations to five and no more than three stations are allowed in any district.

Stations	0	1	2	3
District	-	-	-	-
1	2	0.9	0.3	0.2
2	0.5	0.3	0.2	0.1
3	1.5	1.0	0.7	0.3

Expected Property Damage in Millions of Dollars

(a) Letting $u(k)$ be the number of stations assigned to district k, find the optimal allocation of stations minimizing total expected damage.

(b) Suppose that the budgetary restriction is replaced by the the requirement $3u(0)+u(1)+2u(2) \leq 9$ (in millions of dollars) reflecting differential costs across districts. Now find the optimal allocation of stations.

Figure 1: problem 1 description

0.1.1 part a

Before applying dynamic programming to solve the problem, the solution was first found by brute force in order to verify that the D.P. method when completed was correct. Using brute force, the optimal arrangement is found to be:

> Assign 2 stations to the first district, and 3 stations to third district and no stations are assigned to the second district, for a minimum total expected property damage of 1.1 millions.

The brute force method also generated a list of all the arrangements (44 of them) and the cost of each. For reference, here is the complete table with the small Matlab code used to generated it in the appendix. After this, the graphical D.P. method called branch and bound was used to verify this result.

district 1	district 2	district 3	cost in millions
0	0	0	4.0
0	0	1	3.5
0	0	2	3.2
0	0	3	2.8
0	1	0	3.8
0	1	1	3.3
0	1	2	3.0
0	1	3	2.6
0	2	0	3.7
0	2	1	3.2
0	2	2	2.9
0	2	3	2.5
0	3	0	3.6
0	3	1	3.1
0	3	2	2.8
1	0	0	2.9
1	0	1	2.4
1	0	2	2.1
1	0	3	1.7
1	1	0	2.7
1	1	1	2.2
1	1	2	1.9
1	1	3	1.5
1	2	0	2.6
1	2	1	2.1
1	2	2	1.8
1	3	0	2.5
1	3	1	2.0
2	0	0	2.3
2	0	1	1.8
2	0	2	1.5
2	0	3	1.1*

2	1	0	2.1
2	1	1	1.6
2	1	2	1.3
2	2	0	2.0
2	2	1	1.5
2	3	0	1.9
3	0	0	2.2
3	0	1	1.7
3	0	2	1.4
3	1	0	2.0
3	1	1	1.5
3	2	0	1.9

Let the state x be the number of stations available to be assigned at each stage. For example, if we are at stage 2 and $x = 5$, this means all 5 stations are available to be assigned to district two. Stage one was the decision to decide on district one, stage two for the decision to assign for district two and the final stage, stage three is for district three. This is arbitrary, any order will give the same answer. One can decide on district three first, and then district one for example. This makes no difference to the final result.

The following diagram shows the result found which agrees with the brute force method above. The branch cost is the number above the arrow itself. The number in the small rectangle at the node, is the minimal cost of the branch leaving that node. For example, in stage three, when $x = 5$, there are 4 branches that leave that node where (0,1,2,3) stations can be assigned. The lowest cost of these branches is the one with cost 0.3 and that is what goes in the small square next to the node. This process continues moving backward. This method is the graphical equivalent to the Bellman dynamic equations and can be used when the number of states is finite and the number of decisions at each state is finite also.

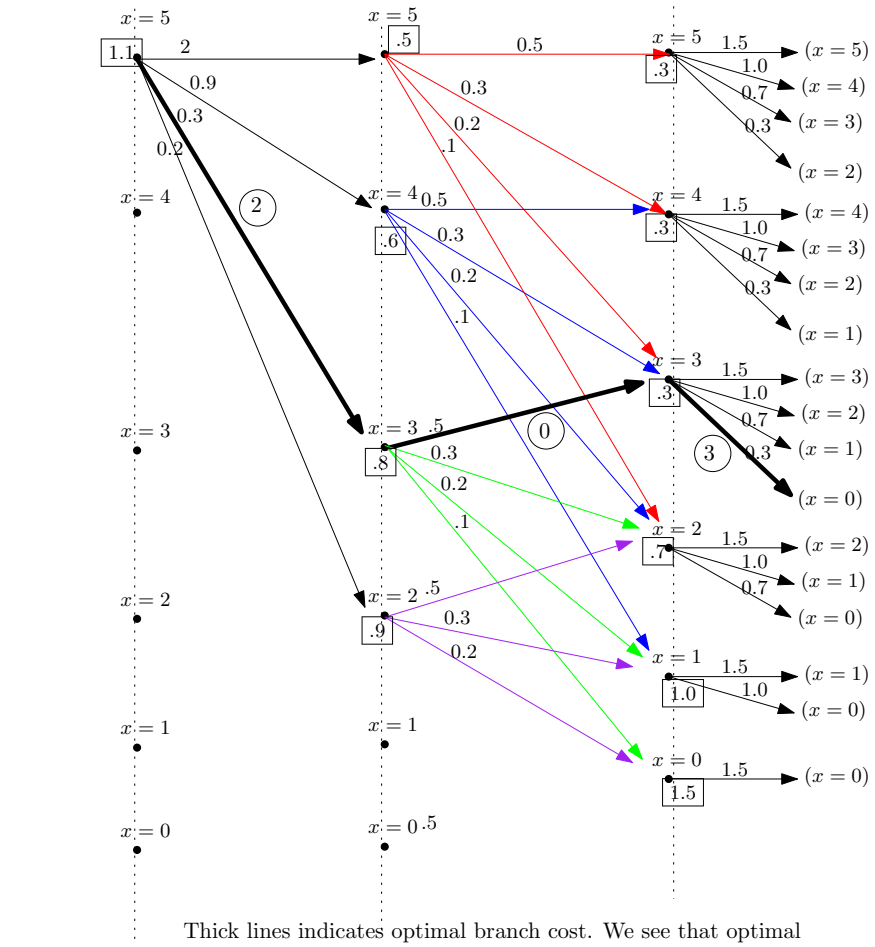

Figure 2: Part (a) solution using Dynamic programming

0.1.2 Part b

The constraint now is

$$3u(0) + u(1) + 2u(2) \leq 9 \tag{1}$$

What this means, is that 3 times the number of stations assigned to district one plus one times the number of stations assigned to district two, plus 2 times the number of stations assigned to district three, must be smaller than 9 stations in total (millions of dollars in the problem was a typo).

We see that part(a) does not meet this requirement. In part(a), we found $u(0) = 2, u(1) = 0, u(3) = 3$, which means

$$\begin{aligned} 3u(0) + u(1) + 3u(2) &= 3(2) + 0 + 2(3) \\ &= 12 \end{aligned}$$

Which is larger than 9. We need to find again $u(0), u(1), u(2)$ which still satisfies part (a) requirements of no more than 3 stations for a district and no more than total of 5 stations, but now with the additional constraint of (1) in place at the same time.

The search was repeated using brute force to first find the combinations that meet this criteria, and then the one with the minimum expected damage was selected.

Here is the new table found

district 1	district 2	district 3	cost in millions	$3u_0 + u_1 + 2u_2$
0	0	0	4.0	0
0	0	1	3.5	2
0	0	2	3.2	4
0	0	3	2.8	6
0	1	0	3.8	1
0	1	1	3.3	3
0	1	2	3.0	5
0	1	3	2.6	7
0	2	0	3.7	2
0	2	1	3.2	4
0	2	2	2.9	6
0	2	3	2.5	8
0	3	0	3.6	3
0	3	1	3.1	5
0	3	2	2.8	7

1	0	0	2.9	3
1	0	1	2.4	5
1	0	2	2.1	7
1	0	3	1.7	9
1	1	0	2.7	4
1	1	1	2.2	6
1	1	2	1.9	8
1	2	0	2.6	5
1	2	1	2.1	7
1	2	2	1.8	9
1	3	0	2.5	6
1	3	1	2.0	8
2	0	0	2.3	6
2	0	1	1.8	8
2	1	0	2.1	7
2	1	1	1.6*	9
2	2	0	2.0	8
2	3	0	1.9	9
3	0	0	2.2	9

We see that the combination with the minimum expected damage is when

> two stations are assigned to district one, and one station assigned to district two and one station assigned to district three with expected damage of 1.6 million dollars.

The following diagram illustrates the branch and bound graph with the new optimal path now highlighted in the think line.

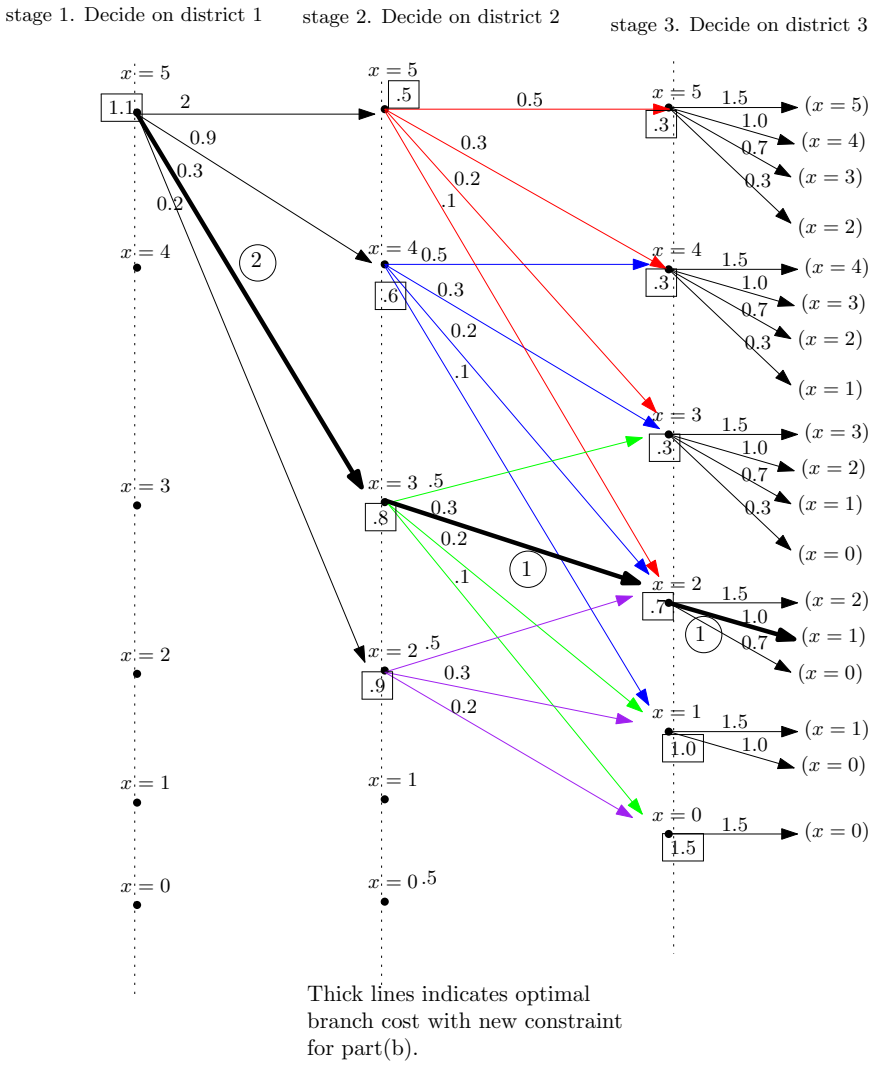

Figure 3: Part (b) solution

The code used to part(b) is in the appendix.

0.1.3 appendix for problem 1

code to generate the first table for part(a)

```
function nma_HW7_ECE_719_prob_1()
%find cost using brute force search, to verify DP method
%HW 7, ECE 719, APril 23,2016
%Nasser M. Abbasi
cost_table   = zeros(100,4); %place to put the cost
count_so_far = 0;
for i=0:3
    j = 0; k = 0;
    if sum([i j k])<=5
      for j=0:3
        k = 0;
        if sum([i j k])<=5
           for k = 0:3
             if sum([i j k])<=5
               count_so_far = count_so_far+1;
               fprintf('count_so_far=%d, [%d,%d,%d]\n', ...
                                        count_so_far,i,j,k);
               cost_table(count_so_far,1:3)=[i j k];
               cost_table(count_so_far,4)=find_cost([i j k]);
             end
           end
        end
      end
    end
end

for i=1:count_so_far
 fprintf('%d & %d & %d & %2.1f \\\\ \\hline\n',cost_table(i,1),...
        cost_table(i,2),cost_table(i,3),cost_table(i,4));
end

[~,J]=min( cost_table(1:count_so_far,4));
fprintf('optimal allocation is \n');
cost_table(J,:)
end
%==========================
function I=find_cost(x)
tbl=[2,.9,.3,.2;
    .5,.3,.2,.1;
    1.5,1,.7,.3];
I= tbl(1,x(1)+1) + tbl(2,x(2)+1) + tbl(3,x(3)+1);
end
```

code to generate the first table for part(b)

```
function nma_HW7_ECE_719_prob_1_part_b()
%finds lowest cost with constraint that
%3*u(0)+u(1)+2*u(2)<=9 to find optinal case using brute force,
%to verify DP method
%HW 7, ECE 719, APril 30,2016
%Nasser M. Abbasi
cost_table   = zeros(100,5); %place to put the cost
count_so_far = 0;
for i=0:3  %this is district 1
    j = 0; k = 0;
    if sum([i j k])<=5
      for j=0:3  %this is district 2
        k = 0;
        if sum([i j k])<=5
           for k = 0:3 %this is district 3
              if sum([i j k])<=5
                %check if 3*i+j+2*k <= 9 first
                if 3*i+j+2*k <=9
                   count_so_far = count_so_far+1;
                   fprintf('count_so_far=%d, [%d,%d,%d], (3*i+j+2*k=%d) \n', ...
                                 count_so_far,i,j,k,3*i+j+2*k);
                   cost_table(count_so_far,1:3)=[i j k];
                   cost_table(count_so_far,4)=find_cost([i j k]);
                   cost_table(count_so_far,5)=3*i+j+2*k;
                end
              end
           end
         end
       end
    end
end

for i=1:count_so_far
  fprintf('%d & %d & %d & %2.1f & %d \\\\ \\hline\n',...
    cost_table(i,1),cost_table(i,2),cost_table(i,3),...
    cost_table(i,4),cost_table(i,5));
end

[~,J]=min( cost_table(1:count_so_far,4));
fprintf('optimal allocation is \n');
cost_table(J,:)
end
%==========================
function I=find_cost(x)
tbl=[2,.9,.3,.2;
    .5,.3,.2,.1;
```

```
    1.5,1,.7,.3];
I= tbl(1,x(1)+1) + tbl(2,x(2)+1) + tbl(3,x(3)+1);
end
```

0.2 Problem 2

Barmish

ECE 719 – Homework Pattern

Similar to the example studied in class, consider the dynamic program described by scalar state equation

$$x(k+1) = x(k) - u(k)$$

with cost function

$$J = \sum_{k=0}^{N-1} [x(k) - u(k)]^2 + u^2(k)$$

to be minimized. Verify that the optimal solution is is of the form

$$u^*(k) = \gamma(k)x(N-k)$$

and find a description of the optimal gain $\gamma(k)$.

Figure 4: problem 2 description

The state equation is (using indices as subscripts from now on, in all that follows as it is easier to read. Therefore $x(N)$ is written as x_N and similarly for $u(N)$)

$$x_{k+1} = x_k - u_k \tag{1}$$

The cost function to minimize is

$$J = \sum_{k=0}^{N-1} (x_k - u_k)^2 + u_k^2 \tag{2}$$

Now we apply Bellman dynamic equation. This diagram shows the overall process.

The branch cost from x_{N-1} with one step to go is

$$I(x_{N-1}, 1) = \min_{u_{N-1} \in \Omega_{N-1}} \{J(x_{N-1}, u_{N-1}) + \Psi(x_N)\}$$

$\Psi(x_N)$ is the terminal cost. Removing the terminal cost from the rest of the computation to simplify things and replacing J in the above from (2) gives

$$\begin{aligned} I(x_{N-1}, 1) &= \min_{u_{N-1} \in \Omega_{N-1}} \{J(x_{N-1}, u_{N-1})\} \\ &= \min_{u_{N-1} \in \Omega_{N-1}} \left\{(x_{N-1} - u_{N-1})^2 + u_{N-1}^2\right\} \end{aligned} \tag{3}$$

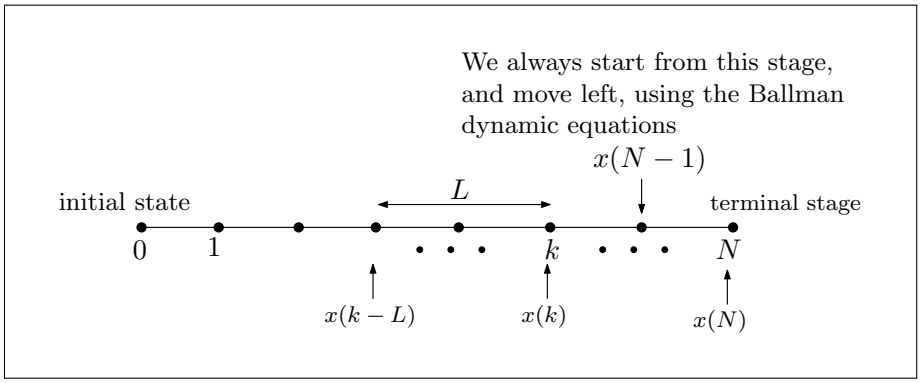

Figure 5: Showing dynamic programming block diagram

Taking derivative in order to find optimal u^*_{N-1} results in

$$\begin{aligned}\frac{dI(x_{N-1},1)}{u_{N-1}} &= 0\\ 2(x_{N-1}-u_{N-1})(-1)+2u_{N-1} &= 0\\ 4u_{N-1}-2x_{N-1} &= 0\\ u^*_{N-1} &= \frac{1}{2}x_{N-1}\end{aligned}$$

Using the above u^*_{N-1} in (3) gives

$$\begin{aligned}I^*(x_{N-1},1) &= \left(x_{N-1}-\frac{1}{2}x_{N-1}\right)^2+\left(\frac{1}{2}x_{N-1}\right)^2\\ &= \frac{1}{2}x^2_{N-1}\end{aligned}$$

Going back one step

$$\begin{aligned}I(x_{N-2},2) &= \min_{u_{N-2}\in\Omega_{N-2}}\{J(x_{N-2},u_{N-2})+I^*(x_{N-1},1)\}\\ &= \min_{u_{N-2}\in\Omega_{N-2}}\left\{(x_{N-2}-u_{N-2})^2+u^2_{N-2}+\frac{1}{2}x^2_{N-1}\right\}\end{aligned}$$

Before taking derivatives, we have to make all the stages to be at $N-2$, and for this we use the state equation to write x_{N-1} in terms of x_{N-2} and the above becomes

$$\begin{aligned}I(x_{N-2},2) &= \min_{u_{N-2}\in\Omega_{N-2}}\left\{(x_{N-2}-u_{N-2})^2+u^2_{N-2}+\frac{1}{2}(x_{N-2}-u_{N-2})^2\right\}\\ &= \min_{u_{N-2}\in\Omega_{N-2}}\left\{\frac{5}{2}u^2_{N-2}-3u_{N-2}x_{N-2}+\frac{3}{2}x^2_{N-2}\right\}\end{aligned}\tag{4}$$

Now we take derivative to find optimal u^*_{N-2}

$$\begin{aligned}\frac{dI(x_{N-2},1)}{u_{N-2}} &= 0\\ 5u_{N-2} - 3x_{N-2} &= 0\\ u^*_{N-2} &= \frac{3}{5}x_{N-2}\end{aligned}$$

We go back to (4) and update with the optimal control found to obtain

$$\begin{aligned}I^*(x_{N-2},2) &= \frac{5}{2}\left(\frac{3}{5}x_{N-2}\right)^2 - 3\left(\frac{3}{5}x_{N-2}\right)x_{N-2} + \frac{3}{2}x^2_{N-2}\\ &= \frac{3}{5}x^2_{N-2}\end{aligned}$$

Going back one more step

$$\begin{aligned}I(x_{N-3},3) &= \min_{u_{N-3}\in\Omega_{N-3}}\{J(x_{N-3},u_{N-3}) + I^*(x_{N-2},2)\}\\ &= \min_{u_{N-3}\in\Omega_{N-3}}\left\{(x_{N-3}-u_{N-3})^2 + u^2_{N-3} + \frac{3}{5}x^2_{N-2}\right\}\end{aligned}$$

Before taking derivatives, we have to make all the states to be at $N-3$, and for this we use the state equation to write x_{N-2} in terms of x_{N-3} and the above becomes

$$\begin{aligned}I(x_{N-3},3) &= \min_{u_{N-3}\in\Omega_{N-3}}\left\{(x_{N-3}-u_{N-3})^2 + u^2_{N-3} + \frac{3}{5}(x_{N-3}-u_{N-3})^2\right\}\\ &= \min_{u_{N-2}\in\Omega_{N-2}}\left\{\frac{13}{5}u^2_{N-3} - \frac{16}{5}u_{N-3}x_{N-3} + \frac{8}{5}x^2_{N-3}\right\}\end{aligned} \tag{5}$$

Now we take derivative to find optimal u^*_{N-3}

$$\begin{aligned}\frac{dI(x_{N-3},3)}{u_{N-3}} &= 0\\ \frac{26}{5}u_{N-3} - \frac{16}{5}x_{N-3} &= 0\\ u^*_{N-3} &= \frac{8}{13}x_{N-3}\end{aligned}$$

We go back to (5) and update the cost to obtain

$$\begin{aligned}I^*(x_{N-3},3) &= \frac{13}{5}\left(\frac{8}{13}x_{N-3}\right)^2 - \frac{16}{5}\left(\frac{8}{13}x_{N-3}\right)x_{N-3} + \frac{8}{5}x^2_{N-3}\\ &= \frac{8}{13}x^2_{N-3}\end{aligned}$$

Let us do one more step backward,

$$\begin{aligned} I(x_{N-4},4) &= \min_{u_{N-4}\in\Omega_{N-4}} \{J(x_{N-4},u_{N-4}) + I^*(x_{N-3},3)\} \\ &= \min_{u_{N-4}\in\Omega_{N-4}} \left\{ (x_{N-4}-u_{N-4})^2 + u_{N-4}^2 + \frac{8}{13}x_{N-3}^2 \right\} \end{aligned}$$

Before taking derivatives, we have to make all the states to be at $N-4$, and for this we use the state equation to write x_{N-3} in terms of x_{N-4} and the above becomes

$$\begin{aligned} I(x_{N-4},4) &= \min_{u_{N-4}\in\Omega_{N-4}} \left\{ (x_{N-4}-u_{N-4})^2 + u_{N-4}^2 + \frac{8}{13}(x_{N-4}-u_{N-4})^2 \right\} \\ &= \min_{u_{N-4}\in\Omega_{N-4}} \left\{ \frac{34}{13}u_{N-4}^2 - \frac{42}{13}u_{N-4}x_{N-4} + \frac{21}{13}x_{N-4}^2 \right\} \end{aligned} \tag{6}$$

Now we take derivative to find optimal u_{N-4}^*

$$\begin{aligned} \frac{dI(x_{N-4},4)}{u_{N-4}} &= 0 \\ \frac{68}{13}u_{N-4} - \frac{42}{13}x_{N-4} &= 0 \\ u_{N-4}^* &= \frac{21}{34}x_{N-4} \end{aligned}$$

We go back to (6) and update to obtain

$$\begin{aligned} I^*(x_{N-4},4) &= \frac{34}{13}\left(\frac{21}{34}x_{N-4}\right)^2 - \frac{42}{13}\left(\frac{21}{34}x_{N-4}\right)x_{N-4} + \frac{21}{13}x_{N-4}^2 \\ &= \frac{21}{34}x_{N-4}^2 \end{aligned}$$

This is so much fun, so let us do one more step backward,

$$\begin{aligned} I(x_{N-5},5) &= \min_{u_{N-5}\in\Omega_{N-5}} \{J(x_{N-5},u_{N-5}) + I^*(x_{N-4},4)\} \\ &= \min_{u_{N-5}\in\Omega_{N-5}} \left\{ (x_{N-5}-u_{N-5})^2 + u_{N-5}^2 + \frac{21}{34}x_{N-4}^2 \right\} \end{aligned}$$

Before taking derivatives, we have to make all the states to be at $N-5$, and for this we use the state equation to write x_{N-4} in terms of x_{N-5} and the above becomes

$$
\begin{aligned}
I(x_{N-5},5) &= \min_{u_{N-5}\in\Omega_{N-5}} \left\{ (x_{N-5}-u_{N-5})^2 + u_{N-5}^2 + \frac{21}{34}(x_{N-5}-u_{N-5})^2 \right\} \\
&= \min_{u_{N-5}\in\Omega_{N-5}} \left\{ \frac{89}{34}u_{N-5}^2 - \frac{55}{17}u_{N-5}x_{N-5} + \frac{55}{34}x_{N-5}^2 \right\} \qquad (7)
\end{aligned}
$$

Now we take derivative to find optimal u_{N-5}^*

$$
\begin{aligned}
\frac{dI(x_{N-5},5)}{u_{N-5}} &= 0 \\
\frac{89}{17}u_{N-5} - \frac{55}{17}x_{N-5} &= 0 \\
u_{N-5}^* &= \frac{55}{89}x_{N-5}
\end{aligned}
$$

We go back to (7) and update to obtain

$$
\begin{aligned}
I^*(x_{N-5},5) &= \frac{89}{34}\left(\frac{55}{89}x_{N-5}\right)^2 - \frac{55}{17}\left(\frac{55}{89}x_{N-5}\right)x_{N-5} + \frac{55}{34}x_{N-5}^2 \\
&= \frac{55}{89}x_{N-5}^2
\end{aligned}
$$

Summary table of finding

k	$u^*(N-k)$	$I^*(x_{N-k},k)$
1	$\frac{1}{2}x_{N-1}$	$\frac{1}{2}x_{N-1}^2$
2	$\frac{3}{5}x_{N-2}$	$\frac{3}{5}x_{N-2}^2$
3	$\frac{8}{13}x_{N-3}$	$\frac{8}{13}x_{N-3}^2$
4	$\frac{21}{34}x_{N-4}^2$	$\frac{21}{34}x_{N-4}^2$
5	$\frac{55}{89}x_{N-5}$	$\frac{55}{89}x_{N-5}^2$

This is generated using bisection of the Fibonacci sequence, let $\gamma(k) = \frac{\beta(k)}{\alpha(k)}$ where[1]

$$
\begin{aligned}
\beta(k) &= 3\beta(k-1) - \beta(k-2) \\
\beta(0) &= 0 \\
\beta(1) &= 1
\end{aligned}
$$

And[2]

[1] sequence is `https://oeis.org/A001906`

[2] sequence is `https://oeis.org/A001519`

$$\alpha(k) = 3\alpha(k-1) - \alpha(k-2)$$
$$\alpha(0) = 1$$
$$\alpha(1) = 1$$

Here is program which generates up to $k = 20$

```
Clear[k];
alpha[k_] := alpha[k] = If[k == 0 || k == 1, 1,
                           3*alpha[k - 1] - alpha[k - 2]]
beta[k_] := beta[k] = If[k == 0, 0,
               If[k == 1, 1, 3*beta[k - 1] - beta[k - 2]]];
Table[beta[k]/alpha[k + 1], {k, 1, 20}]
```

.

which gives

$$\frac{1}{2}, \frac{3}{5}, \frac{8}{13}, \frac{21}{34}, \frac{55}{89}, \frac{144}{233}, \frac{377}{610}, \frac{987}{1597}, \frac{2584}{4181}, \frac{6765}{10946}, \frac{17711}{28657}, \frac{46368}{75025}, \frac{121393}{196418}, \frac{317811}{514229}, \frac{832040}{1346269}, \frac{2178309}{3524578}, \frac{5702887}{9227465}, \frac{14930352}{24157817}, \frac{39088169}{63245986}, \frac{102334155}{165580141}$$

or numerically

{0.5, 0.6, 0.6153846153846154, 0.6176470588235294, 0.6179775280898876, 0.6180257510729614, 0.618032786885245

The golden ratio is

$$\varphi = \frac{1+\sqrt{5}}{2} = 1.6180339887482036$$

Therefore in the limit, for large k

$$\boxed{u^*(N-k) = \frac{1}{\varphi} x(N-k)}$$

0.3 Problem 3

Barmish

ECE 719 – Homework Population

A discrete-time system has two populations levels described by the state equations

$$x_1(k+1) = [1 + u^2(k)]x_1(k);$$

and

$$x_2(k+1) = \frac{e^{-u(k)x_1(k)}}{x_1(k)} + 2x_2(k); \quad k = 0, 1, \ldots, N.$$

For the final stage, find the feedback control law $u(N-1)$ minimizing the the total population

$$J = x_1(N) + x_2(N).$$

Figure 6: problem 3 description

Given

$$x_1(k+1) = \left[1 + u^2(k)\right] x_1(k)$$
$$x_2(k+1) = \frac{e^{-u(k)x_1(k)}}{x_1(k)} + 2x_2(k) \qquad k = 0, 1, \cdots, N$$

And the goal is to minimize the objective function at the terminal stage $J = x_1(N) + x_2(N)$. At stage $N-1$ with one step to go

$$I(x(N-1), 1) = \min_{u(N-1)\in\Omega_{N-1}} \{\Psi(x(N))\} \tag{1}$$

Where $\Psi(x(N)) = I(x(N), 0)$. Hence

$$\begin{aligned} I(x(N-1), 1) &= \min_{u(N-1)\in\Omega_{N-1}} \{x_1(N) + x_2(N)\} \\ &= \min_{u(N-1)\in\Omega_{N-1}} \left[1 + u^2(N-1)\right] x_1(N-1) + \frac{e^{-u(N-1)x_1(N-1)}}{x_1(N-1)} + 2x_2(N-1) \end{aligned}$$

Therefore $\frac{dI(x(N-1))}{du(N-1)} = 0$ gives

$$0 = 2u(N-1)\, x_1(N-1) - e^{-u(N-1)x_1(N-1)}$$
$$e^{-u(N-1)x_1(N-1)} = 2u(N-1)\, x_1(N-1)$$

This is of the form $e^{-zx} = 2zx$ where $z \to u(N-1)$ and $x \to x_1(N-1)$, which has root at $z = \frac{0.35173}{x}$ (using Mathematica), hence the control law is

$$\boxed{u^*(N-1) = \frac{0.35173}{x_1(N-1)}}$$

0.4 Problem 4

Barmish

ECE 719 – Homework Floor

For the state equations

$$x_1(k+1) = \min\{x_1(k), x_2(k)\} + u(k)$$

and

$$x_2(k+1) = x_1(k)u(k)$$

with initial condition

$$x_1(0) = 1; \qquad x_2(0) = -1,$$

performance index

$$J = \min_{k=1,2} x_2(k)$$

and control restraint

$$u(k) \in \Omega_k = [-M, M],$$

find the optimal control policy $u^*(0), u^*(1)$ maximizing J.

Figure 7: problem 4 description

The following diagram shows the layout of the stages we need to use. There are three stages. $k = 2$ is the terminal stage, and $k = 0$ is the initial stage.

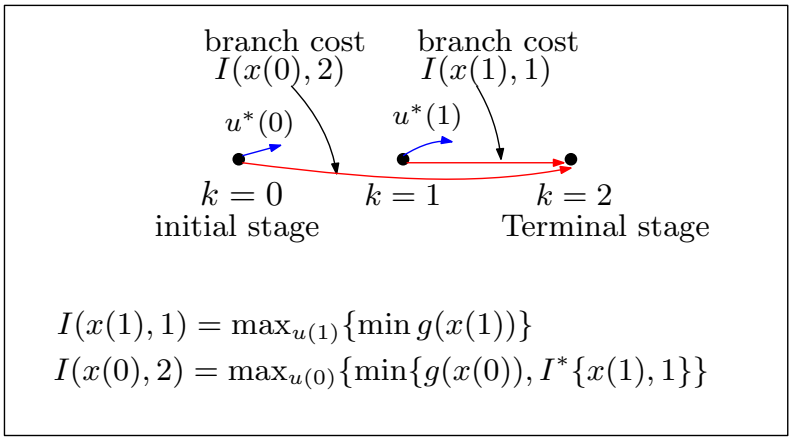

Figure 8: problem 4 stages

We have

$$\begin{aligned} J &= \min_{k=1,2} x_2(k) \\ x_1(k+1) &= \min\{x_1(k), x_2(k)\} + u(k) \\ x_2(k+1) &= x_1(k)\, u(k) \\ x_1(0) &= 1 \\ x_2(0) &= -1 \end{aligned}$$

One step to go, where $N = 2$

$$\begin{aligned} I(x(N-1), 1) &= I(x(2-1), 1) \\ &= I(x(1), 1) \\ &= \max_{u(1)\in\Omega_1} \{x_2(2)\} \end{aligned}$$

But $x_2(2) = x_1(1)\, u(1)$ from the state equation, hence

$$I(x(1), 1) = \max_{u(1)\in\Omega_1} \{x_1(1)\, u(1)\}$$

We need to find $u(1)$ which maximizes $x_1(1)\, u(1)$. Since $u(k) \in \Omega_k = [-M, M]$ then

$$\boxed{u^*(1) = M \text{sign}(x(1))}$$

Therefore

$$\begin{aligned} I^*(x(1), 1) &= x_1(1)\, u^*(1) \\ &= x_1(1)\, M \text{ sign}(x(1)) \\ &= M |x_1(1)| \end{aligned}$$

Now we go one more step backward

$$I(x(0), 2) = \max_{u(0)\in\Omega_0} \min\{x_2(1), I^*(x(1), 1)\}$$

But from state equation, $x_2(1) = x_1(0)\, u(0)$ and since $x_1(0) = 1$ then $x_2(1) = u(0)$ and the above becomes

$$I(x(0),2) = \max_{u(0)\in\Omega_0} \min\{u(0), M|x_1(1)|\} \tag{1}$$

But

$$\begin{aligned} x_1(1) &= \min\{x_1(0), x_2(0)\} + u(0) \\ &= \min\{1,-1\} + u(0) \\ &= -1 + u(0) \end{aligned}$$

Therefore (1) becomes

$$I(x(0),2) = \max_{u(0)\in\Omega_0} \min\{u(0), M|-1+u(0)|\}$$

Let

$$F(u(0)) = \min\{u(0), M|-1+u(0)|\}$$

We need to find $\max_{u(0)} \min(F(u(0)))$. By making small program and adjusting M, to see all the regions, the following result was found for $u^*(0)$

M range	optimal
$0 \le M \le \sqrt{2}$	$u_0^* = \frac{M}{1+M}$
$\sqrt{2} < M$	$u_0^* = M$

The following is a plot of the small program written showing $F(u(0))$ for first case $0 \le M \le \sqrt{2}$ and another plot showing the case for $\sqrt{2} < M$. No other cases were found. The program allows one to move a slider to adjust M and it finds the maximizing u^* for $F(u)$ at each slider change.

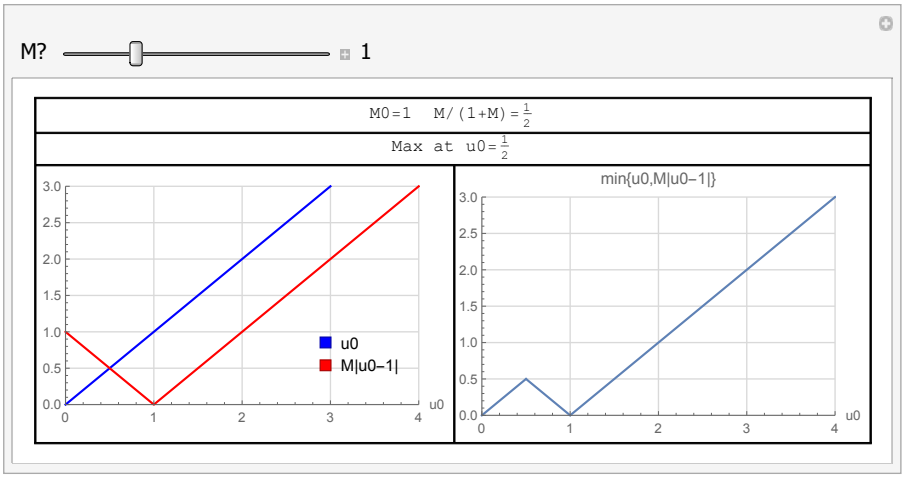

Figure 9: case for $0 \le M \le \sqrt{2}$

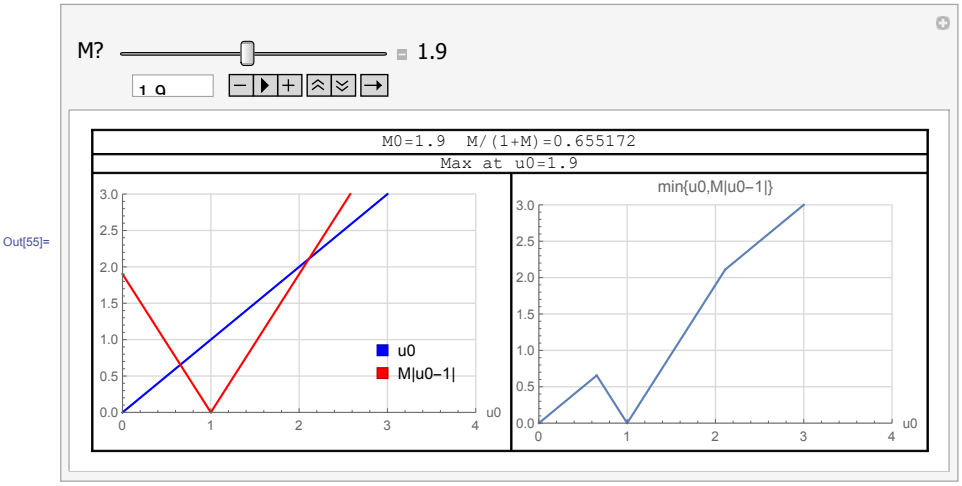

Figure 10: case for $M \geq \sqrt{2}$

Source code for the above

```
Manipulate[
  p0 = Plot[f[u0, M0], {u0, 0, 4}, PlotRange -> {{0, 4}, {0, 3}},
        PlotLabel -> "min{u0,M|u0-1|}",
        ImageSize -> 300, AxesLabel -> {"u0", None},
        GridLines -> Automatic, GridLinesStyle -> LightGray];

  p1 = Plot[u0, {u0, 0, 4}, PlotRange -> {{0, 4}, {0, 3}},
            PlotStyle -> Blue, ImageSize -> 300,
            AxesLabel -> {"u0", None}, GridLines -> Automatic,
            GridLinesStyle -> LightGray];

  p2 = Plot[M0*Abs[u0 - 1], {u0, 0, 4},
        PlotRange -> {{0, 4}, {0, 3}},
        PlotStyle -> Red, ImageSize -> 300];

  u0Max = ArgMax[{Min[u0, M0*Abs[u0 - 1]], u0 >= -M0 && u0 <= M0}, u0];

  p4 = Grid[{{Row[{"M0=", M0, "  M/(1+M)=", M0/(1 + M0)}],
            SpanFromLeft}, {Row[{"Max at u0=", u0Max}],
            SpanFromLeft}, {Legended[Show[p1, p2],
              Placed[SwatchLegend[{Blue, Red}, {"u0", "M|u0-1|"}],
                        {{0.7, 0.1}, {0, 0}}]], p0}},
       Frame -> All];

 p4,

 {{M0, 1, "M?"}, 0, 4, 0.01, Appearance -> "Labeled"},
Initialization :> (
  f[u0_, M0_] := Min[u0, M0*Abs[u0 - 1]]
```

```
)
]
```

.

0.5 Problem 5

Barmish

ECE 719 – Homework Steady State

A discrete time linear system is described by the state equations

$$x_1(k+1) = x_2(k); \quad x_2(k+1) = x_1(k) + u(k)$$

and cost function

$$J = \sum_{k=0}^{\infty} 2x_1^2(k) + 2x_1(k)x_2(k) + x_2^2(k) + 3u^2(k)$$

With feedback control

$$u(k) = K_1 x_1(k) + K_2 x_2(k),$$

find optimal gains K_1 and K_2 minimizing J.

Figure 11: problem 5 description

$$\begin{pmatrix} x_1(k+1) \\ x_2(k+1) \end{pmatrix} = \overbrace{\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}}^{A} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} + \overbrace{\begin{pmatrix} 0 \\ 1 \end{pmatrix}}^{B} u(k)$$

$$J = \sum_{k=0}^{\infty} 2x_1^2(k) + 2x_1(k)\,x_2(k) + x_2^2(k) + 3u^2(k)$$

Since J has the form $x^T Q x + u^T R u$, then we need to find Q first. (Q is symmetric), therefore

$$\begin{aligned} 2x_1^2(k) + 2x_1(k)\,x_2(k) + x_2^2(k) &= \begin{pmatrix} x_1(k) & x_2(k) \end{pmatrix} \begin{pmatrix} q_{11} & q_{12} \\ q_{12} & q_{22} \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} \\ &= \begin{pmatrix} x_1(k)\,q_{11} + x_2(k)\,q_{12} & x_1(k)\,q_{12} + x_2(k)\,q_{22} \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} \\ &= x_1^2(k)\,q_{11} + 2q_{12}x_1(k)\,x_2(k) + x_2^2(k)\,q_{22} \end{aligned}$$

Comparing coefficients, we see that $q_{11}=2, q_{22}=1, q_{12}=1$, hence

$$Q=\begin{pmatrix}2 & 1\\ 1 & 1\end{pmatrix}$$

And R is scalar

$$R=3$$

Therefore, the discrete algebraic Riccati equation is

$$P=APA^T-A^TPB\left(R+B^TPB\right)^{-1}B^TPA+Q \tag{1}$$

Small note: The above is what we had in lecture notes. Matlab has the above in its help pages slightly different. The first term is written as A^TPA instead of APA^T as we had.

Using Matlab

```
>> A=[0,1;1,0];B=[0;1];Q=[2,1;1,1];R=3;
>> [P,L,G] = dare(A,B,Q,R)

P =
    3.7841    1.6815
    1.6815    4.4022
```

$$P=\begin{pmatrix}3.7841 & 1.6815\\ 1.6815 & 4.4022\end{pmatrix}$$

Notice that P is symmetric as expected.

Let us check it is correct first. Substituting P in RHS of (1) gives

$$\begin{aligned}P=&\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}3.7841 & 1.6815\\ 1.6815 & 4.4022\end{pmatrix}\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}^T-\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}^T\begin{pmatrix}3.7841 & 1.6815\\ 1.6815 & 4.4022\end{pmatrix}\begin{pmatrix}0\\ 1\end{pmatrix}\\ &\left(3+\begin{pmatrix}0\\ 1\end{pmatrix}^T\begin{pmatrix}3.7841 & 1.6815\\ 1.6815 & 4.4022\end{pmatrix}\begin{pmatrix}0\\ 1\end{pmatrix}\right)^{-1}\begin{pmatrix}0\\ 1\end{pmatrix}^T\begin{pmatrix}3.7841 & 1.6815\\ 1.6815 & 4.4022\end{pmatrix}\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\\ &+\begin{pmatrix}2 & 1\\ 1 & 1\end{pmatrix}\end{aligned}$$

The above gives $\begin{pmatrix} 3.7841 & 1.6815 \\ 1.6815 & 4.4022 \end{pmatrix}$ which is P. Yes, it satisfies DARE. Now, using

$$\begin{aligned} u^* &= -\left(R + B^T P B\right)^{-1} B^T P A x \\ &= -\left(3 + \begin{pmatrix} 0 \\ 1 \end{pmatrix}^T \begin{pmatrix} 3.7841 & 1.6815 \\ 1.6815 & 4.4022 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}\right)^{-1} \begin{pmatrix} 0 \\ 1 \end{pmatrix}^T \begin{pmatrix} 3.7841 & 1.6815 \\ 1.6815 & 4.4022 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} \\ &= -0.59472 x_1(k) - 0.22716 x_2(k) \end{aligned}$$

Therefore, comparing the above to $u^*(k) = K_1 x_1(k) + K_2 x_2(k)$ we see that

$$K_1 = -0.59472$$
$$K_2 = -0.22716$$

To simulate the result under the new control law u^*, we need some initial condition on state. Below is small script which simulate this up to $k = 20$ stages with the plot generated

```
%simulate x1 and x2 states under optimal control law
%found in problem 5, HW 7. Plot is attached

close all; clear;
A=[0,1;1,0];B=[0;1];
k1=-0.59472; k2=-0.22716;  %found using dare()

N=20;
x=zeros(2,20);
x(:,1)=[1.5;1];   %need non-zero initial state!
for i=2:N
     x(:,i)=A*x(:,i-1)+B*(k1*x(1,i-1)+k2*x(2,i-1));
end
subplot(1,2,1);
plot(1:N,x(1,:),'r',1:N,x(1,:),'r.');
title('x1 using optimal u');
grid;
subplot(1,2,2);
plot(1:N,x(2,:),'b',1:N,x(2,:),'b.');
title('x2, using optimal u');
grid;
```

.

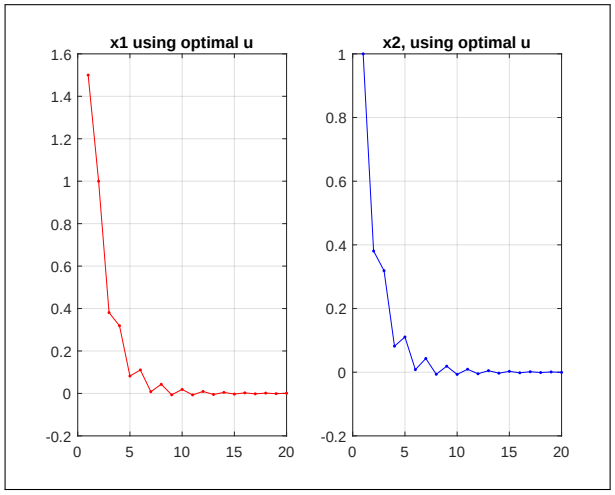

Figure 12: problem 5 plot