The Solution to Homework 7

Problem 1:

In the network pictured below, find the shortest path from node 1 to node 10 and the shortest path from node 2 to node 10.

Solution:

First introduce some notations:

- i: the node number
- t: the time period (the column number of the picture)
- V(i,t) : the shortest path distance from node i in time period t to node 10

The Bellman equation is $V(i,t) = \min_j c_{i,j} + V(j, t+1)$

t = 4

$$V(10,4) = 0$$

t = 3

$$\begin{aligned} V(6,3) &= c_{6,10} + V(10,4) = 3+0 = 3 \\ V(7,3) &= c_{7,10} + V(10,4) = 4+0 = 4 \\ V(8,3) &= c_{8,10} + V(10,4) = 2+0 = 2 \\ V(9,3) &= c_{9,10} + V(10,4) = 3+0 = 3 \end{aligned}$$

t = 2

$$\begin{aligned} V(3,2) &= \min\left(c_{3,6} + V(6,3), c_{3,7} + V(7,3)\right) = \min(6,6) = 6 \\ V(4,2) &= \min\left(c_{4,6} + V(6,3), c_{4,8} + V(8,3), c_{4,9} + V(9,3)\right) = \min(7,4,8) = 4 \\ V(5,2) &= \min\left(c_{5,7} + V(7,3), c_{5,8} + V(8,3)\right) = \min(6,4) = 4 \end{aligned}$$

t = 1

$$\begin{aligned} V(1,1) &= \min\left(c_{1,3} + V(3,2), c_{1,4} + V(4,2)\right) = \min(9,6) = 6 \\ V(2,1) &= \min\left(c_{2,3} + V(3,2), c_{2,4} + V(4,2), c_{2,5} + V(5,2)\right) = \min(10,6,8) = 6 \end{aligned}$$

As a result the shortest paths are $1 \to 4 \to 8 \to 10$ and $2 \to 4 \to 8 \to 10$. Both the shortest distances are 6.

Problem 2

Suppose that a new car costs $20,000 and that the annual operating cost and resale value of the car are as shown in Table below. For example, if you buy a new car and immediately sell it, you will only get $18,000 back. If you don't sell the new car, the operation cost for the 1st year is $600. If I have a new car now, determine a replacement policy that minimizes the net cost of owning and operating a car for the next six years.

Age of Car (Years)	Resale Value ($)	Operating Cost ($)
0	18,000	600 (new)
1	14,000	1,000 (year 1)

Age of Car (Years)	Resale Value ($)	Operating Cost ($)
2	12,000	1,600 (year 2)
3	8,000	2,400 (year 3)
4	6,000	3,200 (year 4)
5	4,000	4,400 (year 5)
6	2,000	6,000 (year 6)

Solution:

First introduce the following notations:

- t: the year. t = 0 means present when I have a new car. t = 1 means the end of first year.
- s: how many years the car has been used at year t.
- NP: new car price.
- RV: resale value.
- OC: operating cost.
- V(t,s): the lowest cost at year t with a car used for s years.

The Bellman equation is

$$V(s,t) = \min(\underbrace{V(s+1,t+1) + OC(s)}_{\text{no replace}}, \underbrace{V(1,t+1) + NP - RV(s) + OC(1)}_{\text{replace}})$$

Because we don't need to own a car after the end of year 6 (t = 6), we will sell the car.

```
NP = 20000
RV = c(18000,14000,12000,8000,6000,4000,2000)
OC = c(600,1000,1600,2400,3200,4400,6000)

T = 7 # because there are 6 years and present

# column j represents the j - 1 th year and
# row i represents the car has been used for i - 1 years at the end of year j
V = matrix(NA, nrow = T, ncol = T)
#tValues = c("Year0","Year1","Year2","Year3","Year4","Year5","Year6")
#sValues = c("Car Age 0","Car Age 1","Car Age 2","Car Age 3","Car Age #4","Car Age 5","Car Age 6")
tValues = 0:6
sValues = 0:6
rownames(V) = sValues # add rownames and colnames to the V matrix
colnames(V) = tValues
names(OC) = sValues
names(RV) = sValues
U = V
t = tValues[T]

V[,paste(t)] = -RV # we will sell the car at year 6 and the cost is the resale value


for(j in rev(tValues[1:T-1])){
  for(i in sValues[1:T-1]){
    V_no_replace = V[paste(i+1),paste(j+1)] + OC[paste(i)]
    V_replace = V[paste(1),paste(j+1)] + NP - RV[paste(i)] + OC[paste(0)]
    V[paste(i),paste(j)] = min(c(V_no_replace,V_replace))
```

```
    U[paste(i),paste(j)] = which.min(c(V_no_replace,V_replace))
  }
}
```

We will keep the car for 2 years. Then replace it and again keep for 2 years. Then keep till the end. The total cost is 8800. (The initial 20,000 cost for buying the car is not included.)

Problem 3:

A sales representative lives in Bloomington and must be in Indianapolis on day 4. On each of the first 3 days, he can sell his wares in Indianapolis, Bloomington, or Chicago. From past experience, he believes that he can earn \$120 from spending a day in Indianapolis, \$160 from spending a day in Bloomington, and \$170 from spending a day in Chicago. Where should he spend the first three days and nights of the week to maximize his sales income less travel costs? Travel costs are shown in Table below.

from/to	Indianapolis	Bloomington	Chicago
Indianapolis	-	50	20
Blomington	50	-	70
Chicago	20	70	-

Solution:

Let us define the sales representative's schedule as following. He leaves one place in the evening and arrives the destination the next morning. Then he will do the sales to make money and leave in the evening again. As a result the initial decision needs to be made at day 0.

First introduce several notations:

- t: day number
- i: place
- Place 1: Indianapolis
- Place 2: Bloomington
- Place 3: Chicago
- TC: traveling costs
- E: earnings
- V(i,t): value function when he is in place i at the begining of t th day.

The Bellman equation is

$$V(i,t) = \max_{j=1,2,3} \left(V(j,t+1) + E_i - TC_{i,j} \right).$$

Because he is required to be Indianapolis on day 4, we can assign $-\infty$ to V(2,4) and V(3,4) in order to avoid them. This is the boundary condition.

t = 3

$$V(1,3) = \max_{j=1,2,3} (V(j,4) + E_1 - TC_{1,j}) = \max(120, -\infty, -\infty) = 120$$
$$V(2,3) = \max_{j=1,2,3} (V(j,4) + E_2 - TC_{2,j}) = \max(110, -\infty, -\infty) = 110$$
$$V(3,3) = \max_{j=1,2,3} (V(j,4) + E_3 - TC_{3,j}) = \max(150, -\infty, -\infty) = 150$$

t = 2

$$V(1,2) = \max_{j=1,2,3}(V(j,3) + E_1 - TC_{1,j}) = \max(240, 180, 250) = 250$$
$$V(2,2) = \max_{j=1,2,3}(V(j,3) + E_2 - TC_{2,j}) = \max(230, 270, 240) = 270$$
$$V(3,2) = \max_{j=1,2,3}(V(j,3) + E_3 - TC_{3,j}) = \max(270, 210, 320) = 320$$

t = 1

$$V(1,1) = \max_{j=1,2,3}(V(j,2) + E_1 - TC_{1,j}) = \max(370, 340, 420) = 420$$
$$V(2,1) = \max_{j=1,2,3}(V(j,2) + E_2 - TC_{2,j}) = \max(360, 430, 410) = 430$$
$$V(3,1) = \max_{j=1,2,3}(V(j,2) + E_3 - TC_{3,j}) = \max(400, 370, 490) = 490$$

Using all the information above, we have

$$V(2,0) = \max_{j=1,2,3}(V(j,1) - TC_{2,j}) = \max(370, 430, 420) = 430.$$

Notice that this is not exactly the Bellman equation above, because we can't make earning on day 0.

Our optimal strategy is to do the sales at Bloomington for the first 3 days and then move to Indianapolis in the evening of day 3.

Problem 4:

A company is considering how to invest \$6,000 and has three investments available. The goal is to maximize Net Present Value (NPV). Let d_j represent the amount invested (in thousands) in investment j. Then $r_j\ (d_j)$ is the net present value for the investment amount d_j in investment j using the following equations:

$$r_1(d_1) = 7d_1 + 2(d_1 > 0)$$
$$r_2(d_2) = 3d_2 + 7(d_2 > 0)$$
$$r_3(d_3) = 4d_3 + 5(d_3 > 0)$$
$$r_1(0) = r_2(0) = r_3(0) = 0$$

Solve this problem to maximize NPV using dynamic programming.

Some Hints: Think of the stages as being the investments. At stage t=1, you make a decision to invest 0, 1, 2, 3, 4, 5, or 6 (in thousands) in investment 1. For example, if you invest $d_1 = 1$ (\$1,000) at t=1, then the NPV for that investment is $r_1(1) = 7 * 1 + 2 = 9$ and there would be \$5,000 remaining for the other investments. If $d_2 = 4$ (\$4,000) is invested in investment 2, the NPV for that investment is $r_2(5) = 3 * 4 + 7 = 19$ and there would be \$1,000 left over for investment 3. The optimal 3rd stage decision is $d_3 = 1$ for $r_3(1) = 4 * 1 + 5 = 9$.

To tackle this, start at the end and work backwards. At stage t=3 (i.e. investment 3), any remaining funds should be fully invested in investment 3 since $r_3\ (d_3\ )$ is non-negative and increasing in d_3. There are 7 possible states to consider at t=3, one for each potential balance remaining for investment. So, you should calculate V(0,3),V(1,3),. . . ,V(6,3) using the notation V(s,t) to represent the optimal value at stage (investment) when the state (remaining balance for investment) is s:

$$V(s = 0, t = 3) = r_3(0) = 0$$
$$V(s = 1, t = 3) = r_3(1) = 4 * 1 + 5 = 9$$
$$\ldots$$
$$V(s = 6, t = 3) = r_3(6) = 4 * 6 + 5 = 29$$

At stage t=2, you should consider the optimal allocation to investment 2 when you have s funds remaining (depending on the stage 1 choices), and considering what the optimal value is in stage 3 after you invest d_2 in investment 2:

$$V(s, t=2) = max_{(d_2)}\{r_2(d_2) + V(s - d_2, t+1 = 3)\}$$

At stage t=1, you will calculate the following for each s:

$$V(s, t=1) = max_{(d_1)}\{r_1(d_1) + V(s - d_1, t+1 = 2)\}$$

Solution:

The Bellman equation is given already, we just need to calculate the values.

```
sValues = (0:6)
tValues = 1:3
sN=length(sValues)
tN=length(tValues)
 slope = c(7,3,4)
  intercept = c(2,7,5)
V = matrix(NA,sN,tN)
U = matrix(NA,sN,tN)
rownames(V) = sValues
colnames(V) = tValues
U = V
#walking backwards in time
for (t in rev(tValues)) {
  #for each time value the loop through the possible investement options
  for (s in sValues){
    if(t==tValues[tN]){
      # Boundary condition
      if (s==0) {V[paste(s), paste(t)] = 0}
      else{
      V[paste(s), paste(t)]= slope[t] * s + intercept[t]}
    U[paste(s),paste(t)] = s
    }
    else {
      #Bellman equation
      value_vec = rep(NA,match(s,sValues))
      names(value_vec) = seq(0,s,by=1)
      for (d in sValues[1:match(s,sValues)]){
        if (d==0) {
          value_vec[paste(d)] = V[paste(s-d),paste(t+1)]
        } else
        {value_vec[paste(d)] = slope[t] * d + intercept[t] + V[paste(s-d),paste(t+1)]}

      }
    V[paste(s),paste(t)] = max(value_vec)
    U[paste(s),paste(t)] = which.max(value_vec) - 1
    }
    }
}
```

The optimal strategy is to invest \$4,000 in investment 1, \$1,000 in investment 2, and remaining \$1,000 in investment 3 to achieve the maximal net present value as \$49,000.

Problem 5:

An electronics firm has a contract to deliver the following number of smart watches during the next three months; month 1, 200 smart watches; month 2, 300 smart watches; month 3, 300 smart watches. For each smart watch produced during months 1 and 2, a \$100 variable cost is incurred; for each smart watch produced during month 3, a \$120 variable cost is incurred. The inventory cost is \$15 for each smart watch in stock at the end of a month. The cost of setting up for production during a month is \$2500. Smart watches made during a month may be used to meet demand for that month or any future month. Assume that production during each month must be a multiple of 100. Given that the initial inventory level is 0 units, use dynamic programming to determine an optimal production schedule.

Solution:

First introduce the following notations:

- t: month number. $t = 1, 2, 3$
- $P(t)$: production of month t.
- $I(t)$: inventory of month t (received from month t-1).
- $D(t)$: demand of month t.
- $V(I(t), t)$: the value function (the optimal cost) of month t with $I(t)$ inventory.

The optimal cost of month t should be the summation of production cost, inventory cost, and the optimal cost of the future. In other words,

$$V(I(t), t) = \min_{P(t)} \text{Production Cost} + \text{Inventory cost} + V(I(t+1), t),$$

where $I(t+1) = P(t) + I(t) - D(t)$. This is the **Bellman equation**. The boundary condition is easy to come up with. No matter how much the inventory is, the cost of month 4 is always 0.

```
InventoryCost = function(I){
  cost = I * 15
  return(cost)
}

ProductionCost = function(P,t){
  if(P == 0){
    cost = 0
  }else{
    if(t %in% c(1,2)){
      cost = 2500 + P * 100
    }else{
      cost = 2500 + P * 120
    }
  }
  return(cost)
}

# initilization
D = c(200,300,300)
```

```r
# add one period to make the boundary conditions easier
time_vals = 1:4
n_time = length(time_vals)
# because total demeand is 8 hundreds, there are 9 possiblity including 0.
inv_vals = seq(0, 800, by = 100)
n_inv = length(inv_vals)

# add names to the matrix
V = matrix(NA,nrow = n_inv, ncol = n_time, dimnames = list(inv_vals, time_vals))
rownames(V) = inv_vals
colnames(V) = time_vals

U = V
InventoryCost = function(I){
  cost = I * 15
  return(cost)
}

ProductionCost = function(P,t){
  if(P == 0){
    cost = 0
  }else{
    if(t %in% c(1,2)){
      cost = 2500 + P * 100
    }else{
      cost = 2500 + P * 120
    }
  }
  return(cost)
}



for(j in rev(time_vals)){
  t = time_vals[j]
  for(i in inv_vals){
    if (j == time_vals[n_time]){V[paste(i),paste(j)]=0} else{
    prod_vals = seq(max(D[j] - i, 0), sum(D) - i + D[j], by = 100)
    n_prod = length(prod_vals)
    value_vec = rep(NA, n_prod)
    names(value_vec) = prod_vals
    for(P in prod_vals){
      value_vec[paste(P)] =  ProductionCost(P,t) + InventoryCost(P + i - D[t]) + V[paste(i + P - D[t]),
    }
    V[paste(i),paste(t)] = min(value_vec)
    U[paste(i),paste(t)] = prod_vals[which.min(value_vec)]
    }
  }
}

# there is no remaining value in month 4
```

The optimal cost is \$89500. We will produce 200 in the first month, 600 in the second month, and produce 0 in the third month.