Home Assignment 2

SF2863 Systems Engineering

Group HA2 35:

Andrew McPherson

andrewmp@kth.se

David Haapanen

davidhaa@kth.se

December 9, 2024

Question 1

By denoting the number of spare parts kept in the local inventory as $\mathbf{s} = [s_1, s_2, \cdots, s_9]$, corresponding to every LRU_1, LRU_2, ..., LRU_9 individually, we let the number of air-crafts grounded due to backorders be denoted by $f(\boldsymbol{s}) = EBO(\boldsymbol{s}) = \sum_{j=1}^{9} EBO_j$ and the total cost of every spare part bought be denoted by $g(\boldsymbol{s}) = C(\boldsymbol{s}) = \sum_{j=1}^{9} c_j s_j$, given that c_j for $j = 1, 2, ..., 9$ represents the cost of buying one spare part corresponding to LRU_j. The total cost can therefore be determined by a function that evaluates the cost of having air-crafts grounded and the cost spent on spare parts. Given that the cost of a grounded aircraft is q, the total cost given the spare parts configuration $\boldsymbol{s}$ is expressed by

$$qf(\boldsymbol{s}) + g(\boldsymbol{s})$$

Thus, in order for it to be possible to optimize this problem using Marginal Allocation, both f and g need to be separable and integer-convex functions and, furthermore, f needs to be strictly decreasing, and g strictly increasing, with respect to $\boldsymbol{s}$. Since f is the sum of EBO_j and g is the sum of $c_j s_j$ for $j = 1, 2, ..., 9$ this means that both are separable, thus proving that each EBO_j and $c_j s_j$ are integer convex will imply that f and g are separable integer-convex functions.

Looking closer at the expression $g_j(s_j) = c_j s_j$ we can see that g_j is strictly increasing with regard to s_j since c_j will always be positive (represents the cost of a spare part). Furthermore, we notice that the forward difference $\Delta g_j(s_j) = c_j(s_j+1) - c_j s_j = c_j$ which gives us that $g_j(s_j) = c_j s_j$ is integer-convex by definition since

$$\Delta g_j(s_j + 1) = c_j(s_j + 2) - c_j(s_j + 1) = c_j \geq \Delta g_j(s_j)$$

Similarly, we can denote $f_j(s_j) = EBO_j(s_j) = \mathbb{E}[(X_j - s_j)^+]$, which represents the expected value of having backorders represented by the random variable $X_j \in Po(\lambda_j T_j)$ minus the number of spare parts corresponding to LRU_j. X_j denotes the number of parts corresponding to LRU_j in the pipeline (number of faulty parts and repaired parts from the workshop that have not yet reached the local inventory). In this case λ_j represents the intensity at which a faulty LRU_j unit arrives and T_j the average amount of hours it takes to repair a broken LRU_j unit. Thus

$$f_j(0) = EBO_j(0) = \mathbb{E}[X_j] = \lambda_j T_j \tag{1}$$

$$f_j(s_j + 1) = EBO_j(s_j + 1) = EBO_j(s_j) - R_j(s_j) \tag{2}$$

where

$$R_j(s_j) = \mathbb{P}(X > s_j) = \sum_{k=s_j+1}^{\infty} \frac{(\lambda_j T_j)^k}{k!} e^{-\lambda_j T_j} = 1 - \mathbb{P}(X \leq s_j) = 1 - \sum_{k=0}^{s_j} \frac{(\lambda_j T_j)^k}{k!} e^{-\lambda_j T_j} \tag{3}$$

The forward difference therefore becomes $\Delta f_j(s_j) = EBO_j(s_j+1) - EBO_j(s_j) = -R_j(s_j)$. Since $R_j(s_j)$ represents a probability it is strictly positive. Thus we can conclude that $\Delta f_j(s_j)$ is strictly negative, meaning f_j is strictly negative. Furthermore $R_j(s_j) \geq 0$ and $R_j(s_j) \geq R_j(s_j + 1)$ meaning by definition that $f_j(s_j)$ is integer-convex since we have that

$$\Delta f_j(s_j + 1) = -R_j(s_j + 1) - (-R_j(s_j)) = R_j(s_j) - R_j(s_j + 1) \geq 0 \geq -R_j(s_j) = \Delta f_j(s_j)$$

Thus $f(\boldsymbol{s}) = \sum_{j=1}^{9} f_j(s_j)$ and $g(\boldsymbol{s}) = \sum_{j=1}^{9} g_j(s_j)$ are strictly decreasing and increasing respectively whilst being separable integer-convex functions, thus meeting required assumptions for applying Marginal Allocation.

Question 2

Since the corresponding costs for each spare part corresponding to LRU_j was given, the vector $\boldsymbol{c} = [c_1, c_2, ..., c_9]$ was known. If there are 2 spare parts for each type then $\boldsymbol{s} = [2, 2, ..., 2]$. Thus the cost of buying all the spare parts was determined by

$$g(\boldsymbol{s}) = \boldsymbol{c}\boldsymbol{s}^T = 538$$

which can be seen in on the screenshot on the last page. To determine the total of air-crafts grounded, expressed as $EBO(\mathbf{s}) = \sum_{j=1}^{9} EBO_j(s_j)$, each $EBO_j(s_j)$ was firstly determined. This was done using the given arrival rates $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_9]$ and repair times $\boldsymbol{T} = [T_1, T_2, ..., T_9]$ corresponding to each LRU in combination with eq. (1), eq. (2) and eq. (3). This was implemented in `MATLAB` such that a function could take a given parts configuration $\boldsymbol{s} = [s_1, s_2, ..., s_9]$ in combination with $\boldsymbol{\lambda}$ and $\boldsymbol{T}$ and return a `1x9` vector corresponding to $[EBO_1(s_1), EBO_2(s_2), ..., EBO_9(s_9)]$

This resulted in the total number of grounded air-crafts, given 2 spare parts of each LRU, to be

$$EBO(\mathbf{s}) = \sum_{j=1}^{9} EBO_j(2) \approx 1.40$$

which is illustrated in the screenshot presented on the last page.

Question 3

All the efficient points that satisfied that a cost would be less or equal to the budget of $C_{budget} = 500$ were determined by implementing the Marginal Allocation algorithm. This was implemented by iteratively deducing efficient points with a function in `MATLAB` until the budget was exceeded. The function would based on the spare parts vector $\boldsymbol{s}$ and the given arrival rates $\boldsymbol{\lambda}$, repair times $\boldsymbol{T}$ and costs for the different spare parts $\boldsymbol{c}$ determine the ratios of $\frac{-\Delta f_j(s_j)}{\Delta g_j(s_j)}$ for $j = 1, 2, ..., 9$. Thus the function would for a specific $\boldsymbol{s}$ determine the vector

$$\left[\frac{-\Delta f_1(s_1)}{\Delta g_1(s_1)}, \frac{-\Delta f_2(s_2)}{\Delta g_2(s_2)}, \ldots, \frac{-\Delta f_9(s_9)}{\Delta g_9(s_9)}\right] = \left[\frac{R_1(s_1)}{c_1}, \frac{R_2(s_2)}{c_2}, \ldots, \frac{R_9(s_9)}{c_9}\right]$$

where $R_j(s_j)$ was determined from eq. (3). Thus the algorithm that was used to determine all the efficient points was as follows

1. The starting spare parts configuration was set to $\boldsymbol{s} = [0, 0, \ldots, 0]$
2. The function would from the given $\boldsymbol{s}$ determine the ratios $\left[\frac{R_1(s_1)}{c_1}, \frac{R_2(s_2)}{c_2}, \ldots, \frac{R_9(s_9)}{c_9}\right]$
3. The value of j that corresponded to the largest ratio of $\frac{R_j}{c_j}$ was identified.
4. The spare parts configuration was altered such that for every element s_k in $\boldsymbol{s}$ for $k = 1, 2, ..., 9$ became:

$$s_k = \begin{cases} s_k + 1, & \text{if } k = j, \\ s_k, & \text{if } k \neq j, \end{cases}$$

5. The total cost for the new spare parts configuration was evaluated as $\boldsymbol{cs}^T$. If this value was below the value of C_{budget} the last spare parts configuration would be stored as an efficient point and the algorithm would start over from 2. If the value exceeded the value of C_{budget} the last spare parts configuration was disregarded and the algorithm terminated.

Question 4
By implementing the algorithm described in **Question 3** all the efficient points given the budget $C_{budget} = 500$ were determined. These can be seen in a table together with the $EBO(\boldsymbol{s})$ value and cost, $C(\boldsymbol{s})$, for every spare parts configuration below in table 1

Table 1: Efficient Points obtained using Marginal Allocation with budget of $C_{budget} = 500$

Efficient Point	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	$EBO(\boldsymbol{s})$	$C(\boldsymbol{s})$
1	0	0	0	0	0	0	0	0	0	7.7120	0
2	0	0	0	1	0	0	0	0	0	6.9148	21
3	0	0	0	1	1	0	0	0	0	6.6289	32
4	1	0	0	1	1	0	0	0	0	6.2943	45
5	1	0	1	1	1	0	0	0	0	5.7436	68
6	1	0	1	2	1	0	0	0	0	5.2699	89
7	1	1	1	2	1	0	0	0	0	4.9490	106
8	1	1	1	2	1	1	0	0	0	4.2654	159
9	1	1	1	2	1	1	1	0	0	3.3514	235
10	1	1	1	3	1	1	1	0	0	3.1359	256
11	1	1	1	3	1	1	2	0	0	2.4329	332
12	1	1	1	3	1	1	2	0	1	2.2223	355
13	1	1	1	3	1	1	2	1	1	1.9307	387
14	1	1	2	3	1	1	2	1	1	1.7395	410
15	1	1	2	3	1	2	2	1	1	1.4200	463

Thus one can see that of the efficient points, the spare parts configuration that leads to the minimum $EBO(\boldsymbol{s})$ value while leading to a cost under the budget is efficient point number 15. This is equivalent to the spare parts configuration of $\boldsymbol{s} = [1, 1, 2, 3, 1, 2, 2, 1, 1]$

All the efficient points were also plotted with regards to the total amount of grounded air-crafts on the vertical axis ($EBO(\boldsymbol{s})$)and the total cost of spare parts, $C(\boldsymbol{s})$, on the horizontal axis. The plot is illustrated below in fig. 1

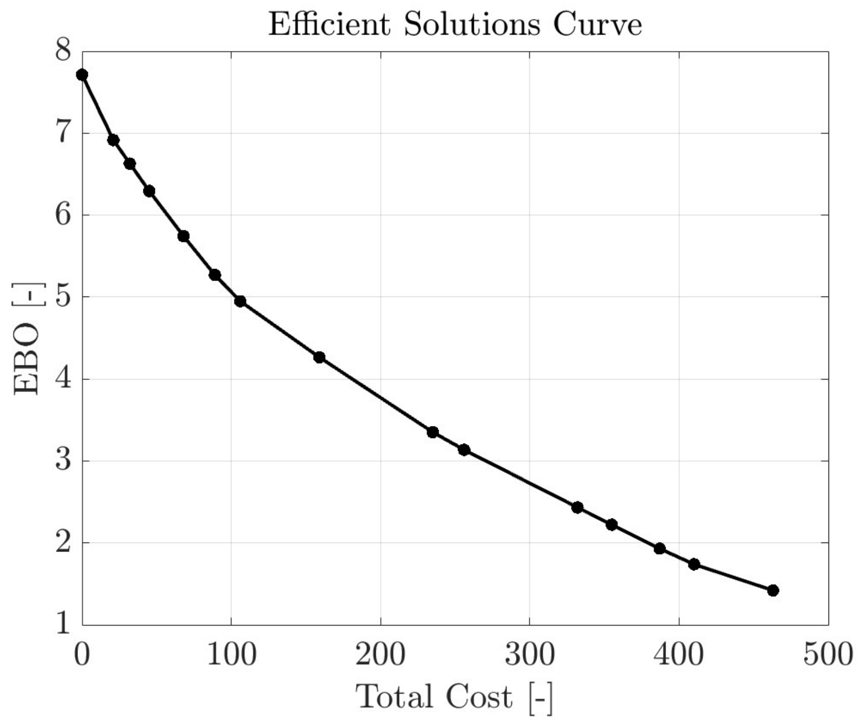

Figure 1: All the efficient points in table 1
plotted with $EBO(\boldsymbol{s})$ in the vertical axis and the total cost in the horizontal axis.

It can be seen in fig. 1 that as more spare parts are bought, hence the closer the cost is to the budget restriction of $C_{budget} = 500$, the value of $EBO(\boldsymbol{s})$ decreases.

Question 5
The best solution with regards to the total $EBO(\boldsymbol{s})$ is the spare parts configuration $\boldsymbol{s}$ that minimizes its' value. If there were two spare parts of every kind, such as in **Question 1** this value would be $EBO([2, 2, ..., 2]) \approx 1.40$ whilst the cost for this configuration would be 538

Considering the Marginal Allocation approach we can see that efficient point 15 in table 1, $\boldsymbol{s} = [1, 1, 2, 3, 1, 2, 2, 1, 1]$, is the spare parts configuration that minimizes $EBO(\boldsymbol{s})$ to 1.42 while costing 463.

Thus the spare parts configuration from **Question 1** would exceed the given budget of 500 with a cost of 38 while only decreasing the total $EBO(\boldsymbol{s})$ with a value of 0.2 when compared to the spare parts configuration of $\boldsymbol{s} = [1, 1, 2, 3, 1, 2, 2, 1, 1]$.

Question 6
In order to solve the problem using Dynamic Programming several things were defined

- **Stages:** Given stage k the spare part corresponding to type LRU_k is considered. Since there are 9 different LRUs the Dynamic Programming algorithm had the same amount of stages. For every stage k the decision of how many spare parts of LRU_k that should be bought was determined.
- **States:** Every stage had a number of different states that correlated to how much

of the budget remained upon entering the current stage. Since each stage could be entered with any value between 0 and $C_{budget} = 500$ there was a total of 501 states which were stated as $C_{remaining} = \{0, 1, 2, ..., 500\}$.

- **Decision variable and state update equation:** While being in stage k at state $C^k_{remaining}$ the decision variable s_k, denoting the amount of spare parts of type LRU_k to be bought, was determined. Hence this gave an expression for how to update each state when moving to the next stage $k+1$. Given that every cost c_k for every type of LRU_k is given the state update equation was determined as
$$C^{k+1}_{remaining} = C^k_{remaining} - s_k c_k$$
Note that since a budget is given there exists a maximum value of s_k dependent on the cost of LRU_k and $C^k_{remaining}$. This gives the feasible values of s_k such that $s_k \in F = \{0, 1, ..., \left\lfloor \frac{C^k_{remaining}}{c_k} \right\rfloor\}$. Here $\lfloor x \rfloor$ denotes the so called floor function which outputs the greatest integer less than or equal to x.

- **Value function:** The value function that was implemented in the algorithm in order to determine the values for the decision variable s_k at every stage s_k given the state $C^k_{remaining}$ was the minimum EBO value that could be achieved from stage k onward. Thus the value function was stated as
$$V_k(C^k_{\text{remaining}}) = \min_{s_k \in F} \left\{\text{EBO}_k(s_k) + V_{k+1}(C^{k+1}_{\text{remaining}})\right\}$$
were $EBO_k(s_k)$ denotes the expected number of grounded air-crafts caused by having s_k number of spare parts of type LRU_k. Thus the value of $V_k(C^k_{remaning})$ is determined by the minimum value of $EBO_k(s_k)$ whilst taking into account for the minimum EBO value from remaining stages given a choice for $s_k \in F$.

- **Recursive relation of value function:** For each stage $k = 1, 2, ..., 9$ given the entering state $C^k_{remaining}$ the value function was determined using the expression of the value function, which in itself depends on the value function given the next stage $k+1$ and the updated state $C^{k+1}_{remaining}$. Thus in order to evaluate the value function in stage k, the value functions different values are needed for all later stages $k+1, ..., 9$

  Thus it was set that $V_{k=10}(C_{remaining}) = 0$ for all states $C_{remaining} = \{0, 1, 2, ..., 500\}$, since we only consider 9 LRU, which then allowed the Dynamic Programming algorithm to determine $V_k(C^k_{remaining})$ for all stages and states.

Given all of the previously listed things an algorithm using Dynamic Programming was constructed in order to evaluate the optimal spare parts configuration $\boldsymbol{s} = [s_1, s_2, ..., s_9]$ for a given budget value C_{budget}. The algorithm can be described by the following steps

1. Stage $k = 9$ is considered first were the entering state can be one of many given as $C^9_{remaining} = \{0, 1, ..., C_{budget}\}$. For each possible $C^9_{remaining}$ the value $V_9(C^9_{remaining})$ is determined by the minimum value of $EBO_9(s_9)$ evaluated for every feasible decision $s_9 \in F = \{0, 1, ..., \left\lfloor \frac{C^9_{remaining}}{c_9} \right\rfloor\}$. Thus for a given $C^9_{remaining}$ both $V_9(C^9_{remaining})$ and the corresponding s_9 was determined and stored. This was done for all possible values of $C^9_{remaining}$.

2. Let the new considered stage have the value of the previous stage minus one. Thus the stage $k-1$ is considered were the possible entering states are $C^{k-1}_{remaining} = \{0, 1, ..., C_{budget}\}$. For each one of these values $V_{k-1}(C^{k-1}_{remaining})$ is determined by the minimum value of $(EBO_{k-1}(s_{k-1}) + V_k(C^k_{remaining}))$, were $C^k_{remaining}$ is given by the state update equation, for every feasible decision $s_{k-1} \in F = \{0, 1, ..., \left\lfloor \frac{C^{k-1}_{remaining}}{c_{k-1}} \right\rfloor\}$. This was done for each $C^{k-1}_{remaining}$ letting both $V_{k-1}(C^{k-1}_{remaining})$ and the corresponding s_{k-1} be determined and stored each time.

3. Perform step 2. until the stage that is considered has the value 0, when that has happened go directly to step 4.

4. From the given budget C_{budget} and all previously stored values find $V_1(C_{budget})$ and the corresponding decision value s_1. From this use the state update equation to determine $C^2_{remaining} = C_{budget} - s_1 c_1$. Let this now be the entering state for the next stage k = 2 in order to find $V_2(C^2_{remaining})$ and its corresponding s_2 before once again the state update equation determine $C^3_{remaining}$. Continue this procedure until every value for the spare parts configuration $\boldsymbol{s} = [s_1, s_2, ..., s_9]$ is determined.

5. Terminate the algorithm letting the spare parts configuration $\boldsymbol{s} = [s_1, s_2, ..., s_9]$ be the best option given C_{budget}. The value of $EBO(\boldsymbol{s})$ is then given by $V_1(C_{budget})$ and the cost for the spare parts can be determined by the computation $\boldsymbol{cs}^T$

Question 7
Given that we only consider spare parts of type LRU_1 given the budget $C_{budget} = 50$ we can deduce that the only states that are of interest are $C^1_{remaining} = \{0, 1, 2, ..., 50\}$. Since only LRU_1 is considered the value function in the next stage was set to zero such that $V_2(C^2_{remaining}) = 0$ for every value $s_1 \in F = \{0, 1, ..., \left\lfloor \frac{C^1_{remaining}}{c_1} \right\rfloor\}$. Thus value of $V_1(C^1_{remaining})$ was determined using the value function equation

$$V_1(C^1_{\text{remaining}}) = \min_{s_1 \in F} \left\{ \text{EBO}_1(s_1) + V_2(C^2_{\text{remaining}}) \right\} = \min_{s_1 \in F} \{ \text{EBO}_1(s_1) \}$$

It was then determined that the value $s_1 = 3$ gave the lowest value of $EBO_1(s_1) \approx 0.0009$ while leading to a of $c_1 s_1 = 39 \leq C_{budget} = 50$.

Question 8
The optimization problem was solved when considering all types of LRU for all viable budgets of $C_{budget} = \{0, 1, ..., 500\}$, thus giving several optimal spare parts configurations. This was done by implementing the Dynamic Programming algorithm described in **Question 6**. The results when the budget is only $C_{budget} = \{0, 100, 150, 350, 500\}$ can be seen below in table 2

Table 2: Optimal Allocations $\boldsymbol{s} = [s_1, ..., s_9]$ and corresponding values for $EBO(\boldsymbol{s})$ and cost $C(\boldsymbol{s})$ for budgets of $C_{budget} = \{0, 100, 150, 350, 500\}$

C_{budget}	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	$EBO(\boldsymbol{s})$	$C(\boldsymbol{s})$
0	0	0	0	0	0	0	0	0	0	7.71200000000000	0
100	1	0	1	2	2	0	0	0	0	5.22448592440900	100
150	1	1	1	3	1	0	0	0	1	4.52278482166815	150
350	1	1	1	2	1	1	2	1	0	2.35690972347718	343
500	2	1	2	3	1	1	3	1	1	1.23182977648002	499

It can be seen in table 2 that the spare parts configuration that leads to a minimum of $EBO(\boldsymbol{s}) \approx 1.23$ while having a cost of $C(\boldsymbol{s}) = 499 \leq C_{budget} = 500$ is given by $\boldsymbol{s} = [2, 1, 2, 3, 1, 1, 3, 1, 1]$.

Question 9

The solutions using Dynamic Programming for the given budget of $C_{budget} = 500$ gives more spare parts configuration when compared to the efficient point using Marginal Allocation determined in **Question 4**. These were therefore compared with each other in a single plot with $EBO(\boldsymbol{s})$ in the vertical axis and the total cost for each different spare part configuration $\boldsymbol{s}$ in the horizontal axis. The efficient points given in table 1 have the color blue while the different spare parts configurations determined in **Question 8** have the color black, these are illustrated in fig. 2

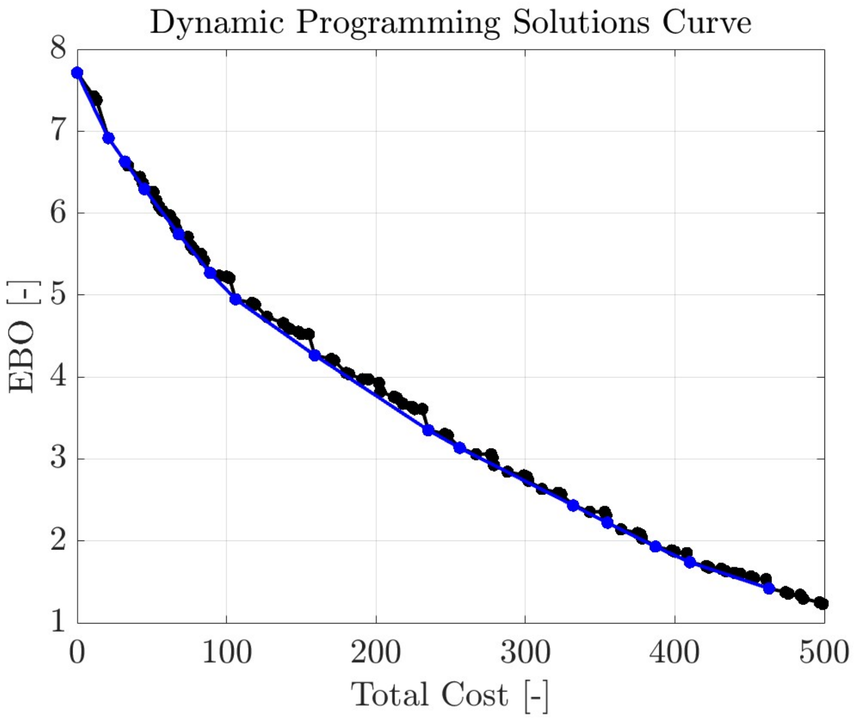

Figure 2: Optimal allocations from Dynamic Programming (black) plotted with $EBO(\boldsymbol{s})$ in the vertical axis and the total cost in the horizontal axis, compared to efficient points from fig. 1 (blue).

We can see in fig. 2 that the efficient points given in table 1 are also viable spare parts configurations that the Dynamic Programming algorithm found. These points show were $EBO(\boldsymbol{s})$ is decreased most significantly when compared to the increase of cost, thus these point decreases $EBO(\boldsymbol{s})$ optimally for a cost increase. The Dynamic Programming algorithm however also yields other spare parts allocation that for a given budget gives the lowest possible $EBO(\boldsymbol{s})$. Yet for most of these points the decrease in $EBO(\boldsymbol{s})$ is not larger than the increase in cost, thus these are not efficient points which can be seen in table 2 as they do not intersect with the blue efficient curve. This result is reasonable since the Dynamic Programming algorithm is designed to find the configuration of the spare parts $\boldsymbol{s}$ that minimizes the $EBO(\boldsymbol{s})$ for a given C_{budget}, without taking into account for the implied increase in cost. The Marginal Allocation does however take this into account and thus only provides the efficient points for given C_{budget}.

Question 10

Given that there would be no budget restriction we can at most buy five spare parts for every type of LRU. Thus for each LRU_j we can either buy 0, 1, 2, 3, 4 or 5 spare parts of type j, thus giving a total of 6 options. Since we have 9 different LRUs we can determine how many configurations of spare parts we can obtain by the following computation

$$6^9 = 10077696$$

Screenshot of MATLAB Grader

```
EBO2 =
    0.0092    0.0080    0.0581    0.3242    0.0054    0.1478    0.8360    0.0058    0.0020
Cost2 =
   538
EPtable =
         0         0         0         0         0         0         0         0         0    7.7110         0
         0         0         0    1.0000         0         0         0         0         0    6.9140   21.0000
         0         0         0    1.0000    1.0000         0         0         0         0    6.6281   32.0000
    1.0000         0         0    1.0000    1.0000         0         0         0         0    6.2934   45.0000
    1.0000         0    1.0000    1.0000    1.0000         0         0         0         0    5.7428   68.0000
LRU1 =
  Columns 1 through 30
     0     0     0     0     0     0     0     0     0     0     0     0     0     1     1     1     1     1     1     1     1     1     1     1     1     1     2
  Columns 31 through 51
     2     2     2     2     2     2     2     2     2     3     3     3     3     3     3     3     3     3     3     3     3
DynPtable =
         0         0         0         0         0         0         0         0         0    7.7110         0
    1.0000         0    1.0000    2.0000    2.0000         0         0         0         0    5.2241  100.0000
    1.0000    1.0000    1.0000    3.0000    1.0000         0         0         0    1.0000    4.5226  150.0000
    1.0000    1.0000    1.0000    2.0000    1.0000    1.0000    2.0000    1.0000         0    2.3567  343.0000
    2.0000    1.0000    2.0000    3.0000    1.0000    1.0000    3.0000    1.0000    1.0000    1.2317  499.0000
NumberOfConfigurations =
    10077696
```

Screenshot of outputs from MATLAB Grader.