HOMEWORK 1 FOR MODERN OPTIMIZATION METHODS

R09946006 | 何青儒 | HO, Ching-Ru | 09/21/2020

1. Profit Maximization Problem for Fertilizer Company

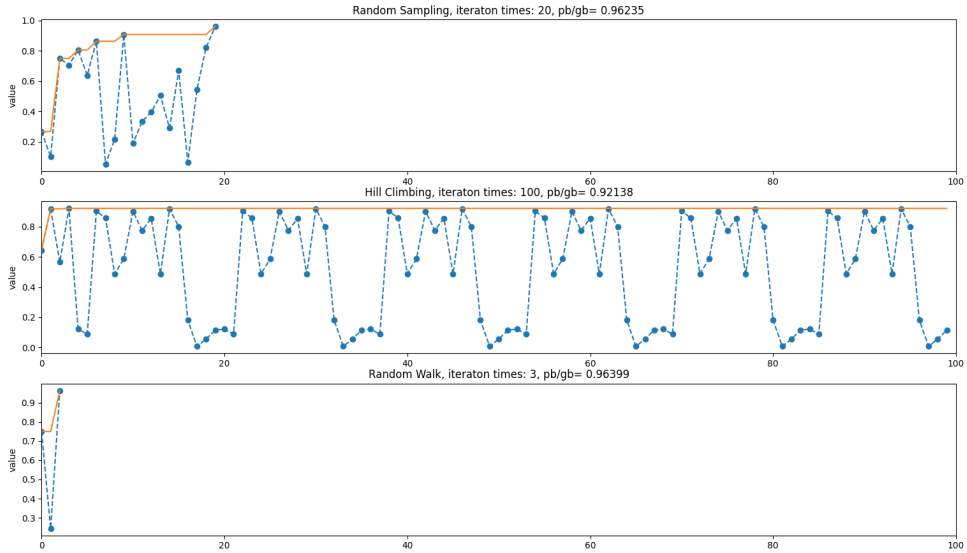
- Denotation (tons):
  - Fertilizer A will be produced $Q_A \in \mathbb{R}$.
  - Fertilizer B will be produced $Q_B \in \mathbb{R}$.
  - Fertilizer C will be produced $Q_C \in \mathbb{R}$.
  - Fertilizer D will be produced $Q_D \in \mathbb{R}$.
- Optimization Problem:
  - Maximize $PROFIT$ where:
  - $$\begin{aligned} PROFIT &= \sum_{i}^{A,B,C,D} (SellingPrice_i - ProductionCost_i - CompositionCost_i) \times Q_i \\ &= (350 - 100 - (1500 \times 5\% + 500 \times 10\% + 1000 \times 5\% + 100 \times 80\%)) \times Q_A \\ &+ (550 - 150 - (1500 \times 5\% + 500 \times 15\% + 1000 \times 10\% + 100 \times 70\%)) \times Q_B \\ &+ (450 - 200 - (1500 \times 10\% + 500 \times 20\% + 1000 \times 10\% + 100 \times 60\%)) \times Q_C \\ &+ (700 - 250 - (1500 \times 15\% + 500 \times 5\% + 1000 \times 15\% + 100 \times 65\%)) \times Q_D \end{aligned}$$
- Subject to:
  - $Q_A \geq 5000$
  - $Q_B \geq 0$
  - $Q_C \geq 0$
  - $Q_D \geq 4000$
  - Nitrates limitation: $5\%Q_A + 5\%Q_B + 10\%Q_C + 15\%Q_D \leq 1000$
  - Phosphates limitation: $10\%Q_A + 15\%Q_B + 20\%Q_C + 5\%Q_D \leq 2000$
  - Potash limitation: $5\%Q_A + 10\%Q_B + 10\%Q_C + 15\%Q_D \leq 1500$

2. Study Time Allocation Problem for a Student

- Denotation:
  - Grade in mathematics course: $g_m \in \{1, 2, 3, 4\}$
  - Grade in statistics course: $g_s \in \{1, 2, 3, 4\}$
  - Student will spend t_m hours in mathematics course and t_s hours in statistics course.
- Optimization Problem:
  - Maximize grade point P where:
  - $P = 4g_m + 3g_s$
- Subject to:
  - $t_s + t_m \leq 40$
  - $t_m : t_s = 75 : 60$, or we can say $t_s = \frac{60}{75} t_m$
  - $\max\{g_m\} = 4, g_m \geq \frac{t_m}{6}$, or we can say $t_m \leq 24$
  - $\max\{g_s\} = 4, g_S \geq \frac{t_s}{5}$, or we can say $t_s \leq 20$

3. Programming of the Traditional Methods

- I used `Python 3.8.5` to finish this assignment.
- 4D matrix saved as `hw1.npy`.
- Load the 4D matrix by `np.load('hw1.npy')`.



Exhaustive Enumeration

- Solution quality: It takes the longest time in all algorithms because it needs to enumerate all elements in 4d matrix (`X` in code). However, it definitely finds the global optima ($= gb = 0.9997407299623859$).
- Time of iterations: 1000 times (enumeration)
- Runtime: 9.385399999999988ms

Random Sampling

- Solution quality: Not bad. It only takes 20 number of iterations to find a solution $pb_{RS} \geq 0.95gb$. But in reality, it might take more time (which means more number of iterations) to reach the expectation.
- Number of iterations: 20 times (due to finding $pb_{RS} \geq 0.95gb$)
- Runtime: 0.09529999999990935ms

Hill Climbing

- Solution quality: Not good. In this case, it traps in the local optima ($pb_{HC}/gb \sim 0.92$).
- Number of iterations: 100 times (due to the number of iterations reaching 100)
- Runtime: 0.28450000000002085ms

Random Walks

- Solution quality: Good, but the test may just lucky. It only takes 3 number of iterations to find a solution $pb_{RW} \geq 0.95gb$. But in reality, it might take more time (which means more number of iterations) to reach the expectation.
- Number of iterations: 3 times (due to finding $pb_{RW} \geq 0.95bg$)
- Runtime: 0.02140000000006026ms

```
PS C:\Users\ching\Documents\MODERN OPTIMIZATION METHODS> python hw1.py
=====Exhaustive_Enumeration=====
Time:  9.385399999999988
X_bestval_EE (= Global Best):  0.9997407299623859 || X_bestloc_EE:  [3, 9, 6, 7]

=====Random_Sampling=====
Times of iteration:  20 || Time:  0.09529999999990935
X_bestval_RS:  0.9621007194609716 || rate of gb:  0.9623502280407936

=====Hill_Climbing=====
Times of iteration:  100 || Time:  0.28450000000002085
X_bestval_HC:  0.921141557580117 || rate of gb:  0.9213804439224695

=====Random_Walk=====
Times of iteration:  3 || Time:  0.02140000000006026
X_bestval_HC:  0.9637355678357742 || rate of gb:  0.9639855003927205
```