40.305 ADVANCED TOPICS IN STOCHASTIC MODELLING COURSE PROJECT

BASIL R. YAP

Implementation

The simulation was implemented in python using an estimated run of 1,000,000 weeks. The code sample can be found in the appendix and at `https://github.com/Dekatria/201804.university.t7.advanced_stochastic.project`.

The implementation is fully customisable with parameters such as the distribution of the store and behaviour of customers as modular components of the simulation program. The store behaviour is scalable in both stores and distribution, more stores with many more weeks accounted for in the future (as long as the distribution remains discrete). A sample of the store configuration file has been attached in the appendix.

Questions Considered

1. Renewal Time

Suitable Regeneration Time: when the units ordered in the week dips below or equal to the average, in this case: **70 units**.

2. Regenerative Simulations

(1) **The expected number of orders in a week in steady-state**
$f_1(q) = g(q) + g_n$ where $g(q)$ is the the number of stores ordering that week.
Empirical Values: 40.74

(2) **The expected number of orders in a week in steady-state**
$f_2(q) = \sum_{i=1}^{n} g_i$
Empirical Values: 70.00

(3) **The probability that more than 75 orders were made in a week in steady state**
$f_3 = \max\{0, f_2(q) - 75\}$
Empirical Values: 0.4301

(4) **The probability there are no store orders that week in steady state**
$f_4 = \begin{cases} 0 & g(q) > 0 \\ 1 & \text{Otherwise} \end{cases}$
Empirical Values: 0.5782

(5) **Cost function value at steady state** $c(q) = 5 + 0.1(g(q) + q_n) + 0.1(f_2) + 100(f_2)^{0.25}$
Empirical Values: 300.04

Date: April 21, 2018 and, in revised form, April 30, 2018.

main python file

```
"""
PURPOSE: Generates simulation based on project guidelines
"""

import pandas as pd
import random
import sys
import helper
import time
import os

def main(length, storeFile, noCus=160, cusProb=0.25):
    print("##########␣SIMULATION␣BEGIN␣##########")
    data = pd.read_csv(storeFile)
    lotList = data["Lot"].tolist()
    visitList = helper.updateVisitList([0 for i in range(data.shape[0])], data
    nextVisit = min(visitList)
    if not os.path.exists("data/simulation/"):
        os.makedirs("data/simulation/")
    with open("data/simulation/%s.csv" % int(round(time.time(), 0)), "a") as
        output.write("week,␣cumTotOrders,␣cumCusOrders,␣cumStoreOrders,␣cu
        cumTotOrders = 0
        cumCusOrders = 0
        cumStoreOrders = 0
        cumCost = 0
        for i in range(int(length)):
            curCusOrders = helper.getCus(noCus, cusProb)
            curStoreOrders = 0
            if (i+1) == nextVisit:
                qList = []
                for j in range(len(visitList)):
                    if visitList[j] == nextVisit:
                        curStoreOrders += lotList[j]
                        qList.append(lotList[j])
                    else:
                        qList.append(0)
                visitList = helper.updateVisitList(visitList, data
                nextVisit = min(visitList)
                qList.append(curCusOrders)
            else:
                qList = [0 for row in visitList]
                qList.append(curCusOrders)
            curTotOrders = curCusOrders + curStoreOrders
            curCost = helper.getCost(qList)

            cumTotOrders += curTotOrders
            cumCusOrders += curCusOrders
            cumStoreOrders += curStoreOrders
            cumCost += curCost
            count = 0
            for k in qList[:-1]:
                if k > 0:
                    count += 1
            count += qList[-1]
            output.write("%s, %s, %s, %s, %s, %s, %s, %s, %s, %s\n" %
            print("%s WEEK %s : %s" % (helper.getProgress(i+1, int(le

    print("################ SIMULATION END ################")

if __name__ == "__main__":
    if len(sys.argv) == 3:
        main(sys.argv[1], sys.argv[2])
    elif len(sys.argv) == 4:
        main(sys.argv[1], sys.argv[2], sys.argv[3])
    elif len(sys.argv) == 5:
        main(sys.argv[1], sys.argv[2], sys.argv[3], sys.argv[4])
    else:
        for i in range(len(sys.argv)):
            print(sys.argv[i])
        raise IndexError("Invalid number of parameters")

# python code/main.py 1000000 config/config_file.1.csv
```

helper functions file

```
"""
PURPOSE: provides helper functions that help with the basic functionalities of th
"""

import numpy as np
import random
from copy import deepcopy

def getCus(noCus, cusProb):
    cusDecList = np.random.uniform(size=noCus)
    return len(cusDecList[np.where(cusDecList <= cusProb)])

def updateVisitList(visitList, data):
    newVisitList = deepcopy(visitList)
    nextVisit = min(visitList)
    for i in range(len(visitList)):
        if newVisitList[i] == nextVisit:
            randDet = random.random()
            row = data.iloc[[i]].values.tolist()[0][2:]
            count = 0
            for j in range(len(row)):
                count += row[j]
                if randDet <= count:
                    newVisitList[i] += j + 1
                    break
    return newVisitList

def getCost(qList):
    count = 0
    for i in qList[:-1]:
        if i > 0:
            count += 1
    count += qList[-1]
    return 5 + 0.1*count + 0.1*sum(qList) + 100*sum(qList)**(0.25)

def getProgress(i, n):
    return "[␣%s␣%%␣]" % round(100*i/n, 2)
```

configuration file

Store	Lot	J1	J2	J3	J4	J5	J6
1	40	0	0.125	0.25	0.25	0.25	0.125
2	30	0	0.2	0.2	0.2	0.2	0.2
3	50	0	0	0.25	0.5	0.25	0