exp3BanditRAPP

TRAN THI CAM GIANG

September 29, 2015

BANDIT EXP3 : Exp3 bandit stands for Exponential-weight algorithm for Exploration and Exploitation. It works by maintaining a list of weights for each of the actions, using these weights to decide randomly which action to take next, and increasing (decreasing) the relevant weights when a payoff is good (bad). We further introduce an egalitarianism factor γ in [0,1] which tunes the desire to pick an action uniformly at random. That is, if $\gamma = 1$, the weights have no effect on the choices at any step. #Ref: http://jeremykun.com/2013/11/08/adversarial-bandits-and-the-exp3-algorithm/

The structure of the programme :

1. call the package "dizzysNEWYANN" that includes the function simulating stochastically SEIR model according tho the formula beta of YANN.
2. call the function "getExtRateMETAPOP" in the file "extRateMETAPOP.r". This function simulates the SEIR model of a metapopulation of n subpopulations. After that, this function calculates the survival time of each subpopulation until the end of simulation. We have a censored data. based on the simulation time, we extract the uncensored data and estimate the extinction rate of the metapopulation.
3. call the functions in the file "exp3Bandit.r", it is the main file in this work :

I have written the codes of EXP3 bandit

I have modelled the EXP3 Bandit for our model as follows :

1. Number of round : based on the CPU budget, here the maximum simulation time (e.x 100years) and the sample step, so ROUND = TSCPU/sampleStep
2. K aims : K est le nombre de villes dans une métapopulation. Ici, on a K différente métapopulations qui ont le nombre de villes de 1 à K.
3. Résultat obtenu : c'est une list qui contient deux événement :
   a. les poids de chaque métapopulation à chaque tour
   b. les récompense obtenues de chaque métapopulation à chaque tour.

CALL the programme as follows : Input:

FOR the Bandit algorithm:

TSCPU : CPU simulaiton time in year

TStep : sample interval

we have the number of round = TSCPU/TStep

KsubPOP : nombre de villes maximum

gamma : un facteur d'égalitarisme gamma in [0,1] qui règle le désir de prendre une action uniformément au hasard.

FOR Parametre de simulation de SEIR

N: population size of subpopulation

nbVilles: number of subpopulation in a metapopulation rho : coupling rate, coupling strength between any two subpopulations

probVISITER : probability that an individual from subpopulation i visits subpopulation j

probINFECTER : the probability that a susceptible individual native from i being in contact with another infected individual native from k gets infected.

```
source("exp3Bandit.r")
```

```
## My package is so cool
## so I will print these lines each time you load it
## Loading required package: deSolve
## Loading required package: survival
## Loading required package: KMsurv
```

```
EXP3Bandit(TSCPU=50,TStep=10,KsubPOP=10,gamma=0.5,
                       N=1e5,nbCONTACT0=100,nbCONTACT1=0.10,grain=1090,nbMulCONTACT=1,phiMIN=0.0,phiMAX=
```

```
## $vecReward
##       Round 1     Round 2    Round 3    Round 4    Round 5
## 1  0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
## 2  0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
## 3  0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
## 4  0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
## 5  0.01034554 0.000000000 0.00000000 0.00000000 0.00000000
## 6  0.00000000 0.008982268 0.00000000 0.00000000 0.00898106
## 7  0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
## 8  0.00000000 0.000000000 0.01516949 0.00000000 0.00000000
## 9  0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
## 10 0.00000000 0.000000000 0.00000000 0.01184936 0.00000000
##
## $vecWeight
##    Round 1  Round 2  Round 3  Round 4  Round 5       NA
## 1        1 1.000000 1.000000 1.000000 1.000000 1.000000
## 2        1 1.000000 1.000000 1.000000 1.000000 1.000000
## 3        1 1.000000 1.000000 1.000000 1.000000 1.000000
## 4        1 1.000000 1.000000 1.000000 1.000000 1.000000
## 5        1 1.000517 1.000517 1.000517 1.000517 1.000517
## 6        1 1.000000 1.000449 1.000449 1.000449 1.000899
## 7        1 1.000000 1.000000 1.000000 1.000000 1.000000
## 8        1 1.000000 1.000000 1.000759 1.000759 1.000759
## 9        1 1.000000 1.000000 1.000000 1.000000 1.000000
## 10       1 1.000000 1.000000 1.000000 1.000593 1.000593
```

We can find here, the results are two data.frame:

1. The weight of each aim (each metapopulation) at each round.
2. The rewards obtained of each each metapopulation at each round.