Exercise CLUSTERS

martedì 29 giugno 2021 11:50

n items

$g = n+1$ clusters

AT EACH STEP

REMOVAL an active clusters

$K = \#$ active clusters $P(\text{REMOVAL}) = 1/K$

$m = \#$ items in the selected cluster

REACCOMODATION NOT into the just destroyed cluster

EACH ITEM

INNOVATION ν → insert the item in an empty cluster

HERDING $\frac{n_i}{N}$

n_i → # items in the i-th cluster

N → total number of items in the syste

Y Yule distr.

$$E(Y) = \frac{\rho}{\rho - 1}$$

$\nu = 0.2$ → 5

$\nu = 0.5$ → 2

```
###########################################
# Code for Exercise  - Clusters #
###########################################

# Number of objects
n<-50
# Number of sites
g<-n+1
# Number of steps
T<-2000
# Probability of innovation
u<-0.5
# Initial state
Y<-rep(c(1),times=n)
Y<-c(Y,0)
# Array of results
A<-Y
# Main cycle
for (t in 1:T) {
  # destruction
  # sites with at least one item
  indexp<-which(Y>0)
  # number of non empty clusters
  Kp<-length(indexp)
  # a cluster is selected and removed
  R<-sample(1:Kp,1)  # the command sample(a:b,1) select at random 1 item
between a and b
  irem<-indexp[R]      # position of the selected cluster
  # size of the removed cluster
  m<-Y[irem]
  # the cluster is removed
  Y[irem]<-0
  # empty sites
  index0<-which(Y==0)
  # number of empty sites
  K0<-length(index0)
  # update of active sites
  indexp<-which(Y>0)
  # update of number of non empty clusters
  Kp<-length(indexp)
  #number of items in the system
  N<-n-m  # from the total of n objects, we eliminate the m objects of the
destroyed cluster
  # creation
  for (i in 1:m) {
    # when all the sites are empty, a new site is filled
    # with uniform probability. It never coincides with the
    # previously destroyed site!
    if (N==0) {
      Y[irem]<-1 # since Y[irem]=1 we are sure that the site filled with
uniform probability is not the destroyed one
      index0<-which(Y==0) # we memorize the index of the empty sites
      K0<-length(index0) # we count the empty sites
      F<-sample(1:K0,1) # we select at random the site to be filled
      icre<-index0[F]
      Y[icre]<-1 # we filled the selected site
      Y[irem]<-0 # we return Y[irem] to its previous value
    } # end if on N==0
    if (N>0) {
      rand<-runif(1)
      if (rand<=u) { # innovation
        # a new cluster is created
        # update of empty clusters
        Y[irem]<-1
        index0<-which(Y==0)
        K0<-length(index0)
        # an empty site is selected and filled
        F<-sample(1:K0,1)
        ifill<-index0[F]
        Y[ifill]<-1
        Y[irem]<-0
      } # end if on innovation
      if (rand>u) { # herding
        # number of active clusters
        indexp<-which(Y>0)
        # frequency vector
        prob<-Y[indexp]/N
        # cumulative probability
        cumprob<-cumsum(prob)
        # pointer to selected site
        indexsite<-min(which((cumprob-runif(1))>0))
        indexsite<-indexp[indexsite]
        Y[indexsite]<-Y[indexsite]+1
      } # end if on herding
    } # end if on N>0
    N<-N+1
  } # end for on i
  A<-c(A,Y)
} # end for on t
A<-matrix(A,nrow=T+1,ncol=g,byrow=TRUE)
# Frequencies
norm<-0
f<-rep(c(0),times=n)
for (j in 1:n) {
  f[j]<-length(which(A==j))
  norm<-norm+f[j]
} # end for on j
# normalize
f<-f/norm



# Yule distribution
rho<-1/(1-u)
fth1<-rep(c(0),times=n)
for (j in 1:n) {
  fth1[j] =rho*beta(j,rho+1)
} # end for on j
k<-c(1:n)

#Plot of the results without the logarithm
```

```
dev.new()
plot(k,f,xlab="i",ylab="f(i)",main="u=0.5")
lines(k,fth1)


# Plot of results with the logarithm
dev.new()
plot(log10(k),log10(f),xlab="i",ylab="f(i)",main="u=0.5")
lines(log10(k),log10(fth1))
```