MATH089 Project 5 - Monte Carlo Simulation

Posted: 11/1/21

Due: 11/12/21, 11:55PM

This project investigates models based upon random realizations of some process, an approach known as Monte Carlo simulation. After an introductory example on approximation of π by random throws of a needle against a pattern of parallel lines (Buffon's needle), approaches to grouping connected entities to maximize some goal is considered. This is a generic problem arising in many fields such as allocation of economic resources or social network analysis. The example considered here is that of election district formation in North Carolina by grouping counties, an application relevant to the topical conversation of gerrymandering.

Tasks:

- Buffon needle convergence plot
- Monte Carlo districting

1 Introduction

(In your own words, describe Monte Carlo simulation and gerrymandering. Make reference to the background paper presented in class)

2 Methods

2.1 Buffon needle

(Describe the Buffon needle problem, define Julia functions to find the crossing probability, and provide a plot of the estimation of π)

```
∴ function πBuffon(l,t,n)
    N=Int(trunc(n))
    x=t*rand(N); θ=π*rand(N); c=cos.(θ)
    nlft = sum(x .- l*c .<= 0)
    nrgt = sum(x .+ l*c .>= t)
    p = (nlft+nrgt)/N
    return 2*l/t/p
  end
```

```
πBuffon
```

```
∴ πBuffon(0.5,1,1.0e6)
```

3.1428751021434405

```
∴ πBuffon.(0.5,1,(5:5:20)*1.0e6)'
```

$$[3.143793993467196, 3.139461142889435, 3.140748235108875, 3.141175581820321] \tag{1}$$

```
∴
```

2.2 Graph representations

(A graph is a structure containing vertices and edges. Describe representation of NC counties by a graph)

2.3 North Carolina geographic and demographic data

- ○ This folded code adds a package to Julia that enables the reading of Matlab files. It only needs to be run once.
- ● This folded code reads NC data

```
∴ using MAT
∴ cd("/home/student/courses/MATH089/voting")
∴ data=matread("NCcounties.mat")["Expression1"];
∴ nCounties, nData=size(data)
```

$$\begin{bmatrix} 100 \\ 13 \end{bmatrix} \tag{2}$$

```
∴ pop=data[:,1]; y=data[:,2]; x=data[:,3];
∴ nNeighbors=Int.(trunc.(data[:,4]));
∴ Neighbors=Int.(trunc.(data[:,5:13]));
∴
```

○

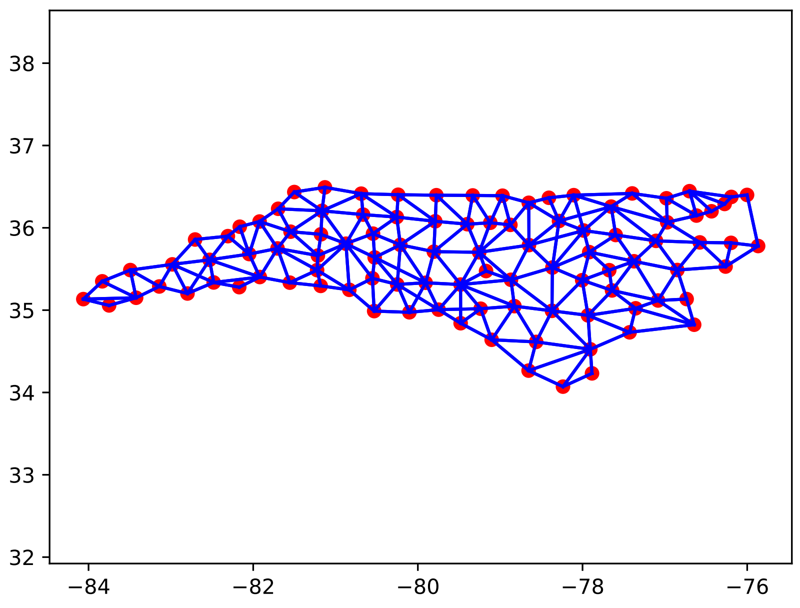

Figure 1. Graph of North Carolina counties. Julia code is folded.

2.4 Combinatorics

Grouping counties into electoral districts is a particular case of the general problem of placing n objects into m groups. Denote by k_i the number of objects in group i. The total number of choices for the first group is known as a combination and given by

$$\binom{n}{k_1} = \frac{n!}{(n-k_1)!\,k_1!}$$

○ Estimation of number of choices is aided by Stirling's formula

$$\log n! = n \log n - n,$$

giving, for instance, the estimate

$$\binom{100}{10} = \frac{100!}{90!\,10!} \cong 1.3 \times 10^{14}.$$

The total number of groupings of counties into districts is given by the sum

$$\sum_{i=1}^{m} \binom{n-\sum_{j=1}^{i-1} k_j}{k_i}.$$

Extra credit: Find the number of possible groupings of North Carolina counties.

2.5 Goal functions

Decisions on whether one districting choice D is better than another requires the definition of quantitative criteria that can be optimized. Many such criteria can be generated, but two simple choices are considered here.

2.5.1 Geographical compactness

The center of gravity of district i is given by the first moments

$$X_i = \frac{1}{k_i}\sum_{j=1}^{k_i} x_{c(j)}, Y_i = \frac{1}{k_i}\sum_{j=1}^{k_i} y_{c(j)},$$

where $c(j)$ is the index of the j^{th} county out of the k_i counties in district i.

How much a district is stretched in the x, y coordinates is given by the centered second moments (variances)

$$A_i = \frac{1}{k_i}\sum_{j=1}^{k_i} (x_{c(j)} - X_i)^2, B_i = \frac{1}{k_i}\sum_{j=1}^{k_i} (y_{c(j)} - Y_i)^2.$$

The ratio A_i/B_i is close to one for an approximately circular district, hence the function

$$f(D) = \sum_{i=1}^{m} \left(\frac{A_i}{B_i} - 1\right)^2,$$

is an overall descriptor of geographical skewness of districting D.

2.5.2 Equal population

Another possible criterion is equal populations in each district, given by the descriptor

$$g(D) = \sum_{i=1}^{m} (p_i - p)^2,$$

with p_i the population in district i, and $p = P/m$ the desired population equal to overall population divided by number of counties.

2.6 Monte Carlo districting

In a Monte Carlo approach, random choices of groupings are attempted, and a record is kept of the grouping that optimizes the goal function. Assume the goal function is

$$F(D) = w f(D) + (1-w) g(D),$$

which should be as small as possible. The weight w determines the relative importance of each criterion.

2.6.1 District generation by random numbers

```
∴ function genD(m,n,v)
    nCD = Int(trunc(n/m)) # Average nr. of counties per district
    D = fill(0,m,2*nCD)   # Allocate space: counties in each district
    nD = fill(0,m)        # Allocate space: nr. of counties in district
    C = collect(1:n)      # List of counties to be distributed
    nC = n                # Nr. of counties that can be distributed
    for i=1:m-1           # Loop over districts
      # Nr. of counties to be placed in district i
      nDi = rand(nCD-v:nCD+v) # Random number close to average
      nDi = min(nDi,nC)   # Cannot be more than available counties
      nDi = max(1,nDi)    # Has to be at least one county
      nD[i] = nDi         # Store number of counties in district
      for k=1:nD[i]       # Loop over nr. of counties to be placed in i
         ic = rand(1:nC)  # Generate a random number to choose county
         D[i,k] = C[ic]   # Place county C[ic] in district i at pos. k
         C[ic:nC-1] = C[ic+1:nC] # Erase county from available counties
         nC = nC - 1      # One less county to distribute
         #@printf("i=%d␣k=%d␣ic=%d␣nC=%d\n",i,k,ic,nC)
         if (nC==0)       # Stop if there are no more counties
           break
         end
      end
      if (nC==0) break; end
    end
    # Last district contains remaining counties
    if (nC>0)
      nD[m] = nC; D[m,1:nC]=C[1:nC]
    end
    return nD,D
  end;
```

```
∴ nD,D=genD(13,100,3)
```

$$\begin{bmatrix} 10 & 21 & 10 & 57 & 67 & 87 & 60 & 40 & 80 & 77 & 8 & 0 & 0 & 0 & 0 \\ 6 & 74 & 1 & 56 & 91 & 33 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 9 & 97 & 45 & 30 & 85 & 66 & 96 & 84 & 58 & 72 & 0 & 0 & 0 & 0 & 0 \\ 4 & 41 & 62 & 17 & 31 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 8 & 5 & 11 & 61 & 2 & 13 & 36 & 19 & 42 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 90 & 92 & 9 & 4 & 95 & 49 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10 & 53 & 43 & 86 & 82 & 44 & 18 & 37 & 63 & 64 & 29 & 0 & 0 & 0 & 0 \\ 8 & 47 & 34 & 23 & 12 & 93 & 59 & 52 & 6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 8 & 26 & 83 & 76 & 39 & 46 & 22 & 35 & 20 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 27 & 99 & 69 & 28 & 79 & 65 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 15 & 89 & 88 & 50 & 14 & 24 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 9 & 3 & 7 & 38 & 25 & 73 & 55 & 98 & 94 & 71 & 0 & 0 & 0 & 0 & 0 \\ 10 & 16 & 32 & 48 & 51 & 54 & 68 & 70 & 75 & 78 & 81 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3}$$

```
∴ sum(nD)
```

100

```
∴ D
```

$$\begin{bmatrix} 21 & 10 & 57 & 67 & 87 & 60 & 40 & 80 & 77 & 8 & 0 & 0 & 0 & 0 \\ 74 & 1 & 56 & 91 & 33 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 97 & 45 & 30 & 85 & 66 & 96 & 84 & 58 & 72 & 0 & 0 & 0 & 0 & 0 \\ 41 & 62 & 17 & 31 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 11 & 61 & 2 & 13 & 36 & 19 & 42 & 0 & 0 & 0 & 0 & 0 & 0 \\ 90 & 92 & 9 & 4 & 95 & 49 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 53 & 43 & 86 & 82 & 44 & 18 & 37 & 63 & 64 & 29 & 0 & 0 & 0 & 0 \\ 47 & 34 & 23 & 12 & 93 & 59 & 52 & 6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 26 & 83 & 76 & 39 & 46 & 22 & 35 & 20 & 0 & 0 & 0 & 0 & 0 & 0 \\ 27 & 99 & 69 & 28 & 79 & 65 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 15 & 89 & 88 & 50 & 14 & 24 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 7 & 38 & 25 & 73 & 55 & 98 & 94 & 71 & 0 & 0 & 0 & 0 & 0 \\ 16 & 32 & 48 & 51 & 54 & 68 & 70 & 75 & 78 & 81 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

```
∴
```

2.6.2 Goal function evaluation

3 Results

3.1 Buffon needle

(Present a log-log plot of the error in estimation of π in the Buffon needle with respect to the number of throws. Estimate te slope of the log-log plot).

3.2 Electoral districting

(Present a plot of the values of the goal function for many districting attempts. Present a table of the five best districting choices. Attempt to plot the best districting you've found)

4 Discussion

(Present your conclusions on the use of random numbers to simulate real-life models)